

A Database Publication

Computing *with the* AMSTRAD

No. 3
March 1985
£1

The independent magazine for CPC 464 users

**We guide you through
the Adventure maze**

**Merging programs...
all you need to know**

**Amstrad's disc drive
- a critical appraisal**

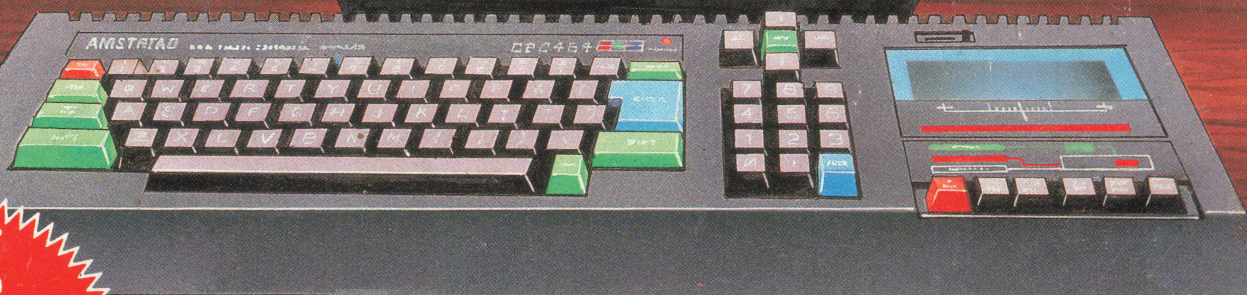
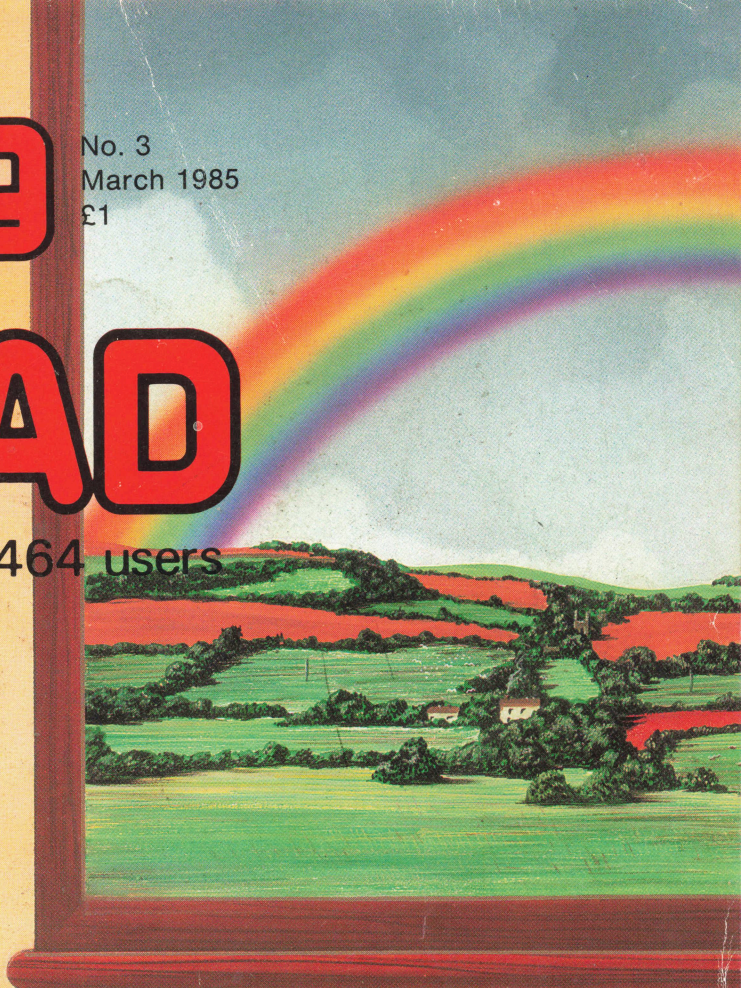
**Go parachute jumping,
fight the swamp frogs,
soar through space...
in 3 all-action games**

**Looking
into
Amstrad's
windows**

FREE
classic 3D
strategy game
with this
month's tape

Amstrad teach-in

**How to handle Basic variables
Machine code byte by byte**





ANIROG

SURVIVOR Search the haunted rooms of Deadstone Abbey for the untold treasures left from years gone by. However, as you help Angus around the ancient building beware of the evil spirits who will chase Angus wherever he goes. All he has to defend himself is his trusty gun and porcupine bombs. Luckily for Angus there are various objects lying around the Abbey such as ammunition, money bags, and bottles of life giving elixir. There are 1008 various rooms all presented in remarkably clear and colourful graphics with beautifully smooth scrolling screens. Ahead lies a terrifying challenge for Angus and its up to you to help him. Are you the sole survivor!

AMSTRAD £7.95

HOUSE OF USHER Enter the House of Usher at your own risk, as you may never leave again. However, once inside there is a choice of nine rooms to select. Behind each door is a totally different action packed arcade game, each of which are certain to strain your nerves to the limit. If you manage to get through these nine rooms another two secret rooms (x and y) will appear, but beware the evil powers of the House of Usher.

AMSTRAD £7.95

FLIGHT PATH Flight Path is without doubt the best flight simulator on the C/16 and Amstrad. The many elaborate features include; Altometer, flaps, directional headings, crosswinds, fires, ground warning lights and reverse thrust to name but a few. Also included are smooth graphics as you take off, cruise over mountains, and land once again.

AMSTRAD £6.95

3D TIME TREK As sole survivor of the planet "Corillian" your quest is one of anger and revenge. The starship you are flying is full of the latest inboard computers and extra powerful sensors. Also included are full 3D graphics, to add unbelievable realism to this fantastic journey through time itself, and beyond.

AMSTRAD £7.95

MOON BUGGY You must skillfully manoeuvre your jumping patrol vehicle over dangerous moon craters as well as large boulders and cunningly placed mines. Not only this but avoid the hovering alien spaceship as it bombards you from above.

AMSTRAD £7.95

Mail Order: 8 HIGH STREET HORLEY SURREY 24 HOUR CREDIT SALES HORLEY 02934 6083 Payment by: P.O. - ACCESS - VISA

AVAILABLE FROM YOUR COMPUTER STORE

A graphic adventure game for the Amstrad

MISSION-1

**AVAILABLE
NOW**

MISSION 1

The Russians have gained a lead in the arms race.

Your Mission is to find their computer, break the code and DELAY the advance.

The stability of the West is in your hands –
GOOD LUCK.



PROJECT VOLCANO

- ★ **Stunning full screen graphics**
- ★ **Loading screen**
- ★ **£8.95 from your usual supplier**
- ★ **Also available for the CBM 64 & Spec 48**

In case of difficulty send £8.95 (inc. post/pack) to:

R&B Software Marketing Ltd,
11 Nuthall Road,
Southport, Merseyside PR8 6XB.

A Database Publication
Computing with the AMSTRAD
 The independent magazine for CPC4000 users

We guide you through the Adventure maze

Merging programs... all you need to know

Amstrad's disc drive - a critical appraisal

Go parachute jumping, fight the swamp frogs, soar through space... in 3 all-action games

Looking into Amstrad's windows

FREE
 Classic 50 strategy game with this month's tape

Amstrad teach-in
 How to handle Basic variables
 Machine code byte by byte

Vol. 1 No. 3 March 1985

Managing Editor: **Derek Meakin**

Features Editor: **Peter Bibby**

The A Team: **Mike Bibby**

Alan McLachlan

Kevin Edwards

Roland Waddilove

Production Editor: **Peter Glover**

Layout Design: **Heather Sheldrick**

News editor: **Mike Cowley**

Advertisement Manager: **John Riding**

Advertising Sales: **Margaret Clarke**

Editor in Chief: **Peter Brameld**

Editorial: 061-456 8835

Administration: 061-456 8383

Advertising: 061-456 8500

Subscriptions: 061-480 0171

Telex: 667664 SHARET G

Prestel Mailbox: 614568383

Published by:
Database Publications Ltd,
Europa House, 68 Chester Road,
Hazel Grove, Stockport SK7 5NY.

Subscription rates for
 12 issues, post free:

£12 - UK

£15 - Eire (Sterling only)

£20 - Rest of world (surface)

£40 - Rest of world (airmail)



Member of Audit
 Bureau of Circulations

"Computing with the Amstrad" welcomes program listings and articles for publication. Material should be typed or computer-printed, and preferably double-spaced. Program listings should be accompanied by cassette tape or disc. Please enclose a stamped, self-addressed envelope, otherwise the return of material cannot be guaranteed. Contributions accepted for publication by Database Publications Ltd will be on an all-rights basis.

© 1985 Database Publications Ltd. No material may be reproduced in whole or in part without written permission. While every care is taken, the publishers cannot be held legally responsible for any errors in articles, listings or advertisements.

"Computing with the Amstrad" is an independent publication and neither Amstrad Consumer Electronics plc or Amsoft are responsible for any of the articles in this issue or for any of the opinions expressed.

News trade distribution:

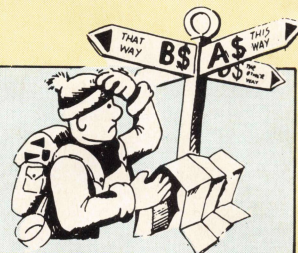
Europress Sales and Distribution Limited, 11 Brighton Road, Crawley, West Sussex RH10 6AF. Tel: 0293 27053.

7 NEWS

Keep up to date with the latest happenings and new arrivals in the busy, expanding world of the Amstrad computer.

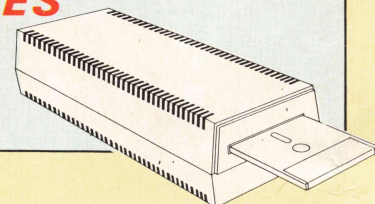
10 BEGINNERS

We're really moving now. Not only are our programs getting longer, we're starting to use string variables.



13 DISC DRIVES

A first look at the Amstrad DDI-1, exploring its AMSDOS, CP/M and LOGO capabilities.



16 ADVENTURE

If you've ever wondered what adventures are all about, why not join Paul James as he comes to grips with Level 9's Colossal Cave? Warning: This article could seriously change your lifestyle.



18 PARACHUTE

Not only a review of "40 Educational Games for the Amstrad" but also a chance to try out one of its programs as you prevent the pilot from crashing to earth.

20 GRAPHICS

This month Michael Noels is looking into windows - text windows, that is. In his usual easy-to-follow style he shows how they can add that professional touch to your programs.



24 BITS AND BYTES

We continue our investigation into the fundamental workings of the Amstrad - 8 bits at a time as we explore the mysteries of "nybbles".

26 MERGE

An in-depth study of how to join two programs together without resorting to super-glue. You'll find these techniques invaluable in developing your own programs.

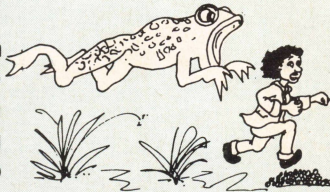
28 MACHINE CODE

Part III of our highly praised series on Z80 machine code. This month we feature a hexadecimal loader and use it to investigate the uses of the A register.



32 SWAMP

Join the Wotawallies in their desperate battle against the giant frogs. Can you survive the unceasing pursuit in your quest to destroy their eggs?



35 STARFLEET

Earth's defences have been seriously weakened by a shortage of qualified starfleet pilots. You've made your application... can you make the grade? Try the Starfleet simulator and find out.



38 SOUND

Still puzzled by the way your micro produces noise and music? Here we delve further into the Amstrad's SOUND facilities with a comprehensive look at the pitch envelope.



44 SOFTWARE SURVEY

More of the latest software releases for the CPC464 assessed by our team of frank and thorough reviewers.

52 ANALYSIS

Ever wished you could line up numbers and achieve that professional finish to your programs? Trevor Roberts dissects a string handling routine which does just that.

54 AL'S BEAT

This month our tame Mr Plod tries his hand at a spot of inventive programming. We always thought he spent his leisure time going round in circles but this...



56 SCREEN DUMP

Short of a screen dump for your Amstrad? Make hard copy of any display in Mode 0 or 1 with this extremely useful utility for Epson compatible printers.

58 EDUCATION

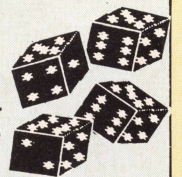
The start of a regular series exploring the growing role of the Amstrad in the classroom.

59 18 COMMANDMENTS

Fancy trying your hand at getting your work of art published in a magazine? Here are 18 hints to show you how it's done.

61 ALEATOIRE

The start of a regular series of puzzles you can put your Amstrad to work on. If only maths at school could have been as much fun...



62 READY REFERENCE

Are you annoyed by Ascii, confused by CHR\$ or aggravated with ASC? Keep cool, the facts are at your fingertips.

63 POSTBAG

Just a small selection from the many interesting and informative letters you've been sending us.

You're never too young to play a Magical Adventure on the Amstrad CPC464...



Based on the style of the classic computer adventures – but written so that even small children can learn to find their way around, encouraged by colourful graphics and exciting sound effects.

The pack contains a 48-page full colour storybook

PLUS

a full length multi-location adventure on cassette for only

£8.95! post free

**Read the book
– then play
the game!**



**Makes
an ideal
present**

Please send me the complete Magic Sword pack for the Amstrad CPC464 containing storybook and cassette to:

Name _____

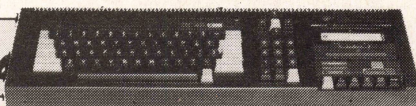
Address _____

- ☐ I enclose my cheque for £8.95 payable to Database Publications
☐ Or debit my Access/Visa card:

No. _____

Signed _____

SEND TO: Adventure offer, Europa House, 68 Chester Road, Hazel Grove, Stockport SK7 5NY



Bang on target!

AMSTRAD hit its target of 200,000 UK sales in 1984 and could have sold "many, many more" according to delighted sales director Dickie Mould.

He said: "Some computer firms count machines they ship to distributors and retailers as sales. These may remain on the shelves.

"We at Amstrad carefully monitor the returned guarantee cards so when we talk of sales we mean real sales to end users and not mere bookkeeping figures.

"I only wish we could have had more machines to send out to the shops".

Now Mould is looking forward to worldwide sales of 600,000 in 1985, "not less than" a third of which he expects to be in the UK.

And the rest? "All over the world" says Mould. "You'd be amazed the number of countries who want the CPC464".

All major outlets where the Amstrad was making its Christmas debut reported record sales receipts over the holiday period.

Try before you buy

A NEW software supplier for the Amstrad CPC464 has notched up a first by launching a try-before-you-buy mail order scheme.

Wolverhampton - based James Paton has just released two games - "Black Phoenix" and "52nd Street" - using this unusual marketing strategy. Both titles will be offered at their usual price of £4.95.

However Paton has not embarked on the project without giving due consideration to the possible pitfalls.

"The main problems associated with distributing software in this manner is that of copyright protection", he told *Computing with the Amstrad*.

"However we feel that the advantages vastly outweigh the disadvantages. And the advantages to the customer are obvious in that they can see what they are buying before they pay a penny".

Daily Mail, Friday, December 28, 1984

Pub brawl millionaires battle it out on the small screen

Zap! Computer gamesmanship

THE MEN REDUCED TO A VIDEO TUSSELE

Chris Curry: Accused

Sir Clive: Apology

Three in search of puppy love

THE pre-Christmas punch-up between computer chiefs Sir Clive Sinclair and Chris Curry is being turned into a video game by a rival company. A firm writing software for big-selling Amstrad home computers plans to launch a game called *This Business Is War*, based on last Friday's pub row. It will feature a bespectacled figure representing Sir Clive and another character resembling Mr Curry slugging it out in a fight to the finish. Instead of using fists the two protagonists will be throwing computers at each other. Mr Chris Anstey, of Amstrad, said last night that it would be obvious who the game was based on, though they would not be named. Sir Clive would be hurling his highly successful Spectrum computer. Mr Curry, boss of Acorn Computers, would be throwing back something resembling the company's popular BBC Micro. Both multi-millionaires figured in a pub brawl in Cambridge after Sir Clive accused Mr Curry of using the £8.95 computer as a weapon.

TYCOONS PUNCH IT OUT!

By JOHN HAMSHIRE

Flashback to the Daily Mail, December 28

Missing cassette foils 'punch up' game coup

A MISSING cassette has cost Amstrad the chance of cashing in on the much publicised clash between computer tycoons Chris Curry and Sir Clive Sinclair.

The pre-Christmas punch-up involving the men behind the BBC Micro and the Spectrum gave Amsoft's whiz kids the bright idea of producing a video game based on the incident.

Called "Business is War", the game would have involved the millionaire computer chiefs continuing their Cambridge pub brawl in a warehouse filled with their products.

It was to have been based on

an existing game, *Electro Freddy*, which has a theme involving Spectrum computers.

Amsoft's aim was to produce the game in time for the recent launch of Sir Clive's revolutionary "electric tricycle".

The plan was for a unit of the SAS - Special Amsoft Services - to stage a daring daylight raid on Sir Clive's press conference armed with free copies of "Business is War".

But the plot flopped even before Amsoft's raiders were fitted with their balaclavas and camouflage jackets.

The cassette containing the original listing of the *Electro*

Freddy program couldn't be found.

"We were almost in tears when we discovered the loss", said Amsoft's William Pole.

"We've missed a wonderful opportunity to promote Amstrad while at the same time taking a dig at two of our competitors.

"We looked everywhere we could think of to find the missing source code without luck.

"And we couldn't contact the author who was away at the time - somewhere in France I believe.

It's very sad".

At Alligata Software, *Electro Freddy's* author Marcus Altman said: "I didn't know about the problem until I got back to work after the holiday.

"I believe I know who has the original listing. But it could not have been ready in time".

Undecided

As we went to press Amsoft was undecided whether to go ahead with production of "Business is War" at a later date.

"Having missed Sir Clive's plastic car launch and with the Sinclair-Curry brawl becoming old news, it may be we'll have to drop the idea", said a disappointed William Pole.

RACE FOR THE HARE

WILL an Amstrad owner win the fabulous Jewelled Hare of Masquerade worth £30,000?

Organisers of the computerised treasure hunt say they expect the prize to be claimed "any time now - in all probability no later than April".

Amstrad users are known to be among individuals and groups claiming to be close to unravelling the secret of where the hare is "buried" within the mind-bending, twin-cassette

puzzle from Haresoft.

"Although the Amstrad only came on the scene just before we issued the first cassette we knew it would be a big seller so we made sure we produced a version for the CPC464" a spokesman said.

Both tapes are needed to find the location of the treasure. To thwart pirates they include information the average computer owner will not be able to reproduce.

CRYSTAL PRIZE

AMSTRAD users who buy Bourne Educational Software's award winning Osprey program from Boots have a chance of winning a valuable crystal sculpture.

The competition – organised by BES, Boots and the Royal Society for the Protection of Birds – is open to individuals and also to groups such as schools and youth organisations.

The first prize in both sections is a 6½in high, 3½ lb specially commissioned Osprey in full lead crystal by Swedish sculptor Mats Jonasson.

Ten runners up will each receive a copy of the Book of British Birds published by the RSPB.

Entrants must submit a project folder – maps, drawings, essays and photos – based on either the history of the Osprey



and its return to Scotland, or a local bird of prey compared to the Osprey.

The competition runs until June 30.

**INTRODUCING
THE CAMSOFT RANGE
OF AMSTRAD DISC BASED
BUSINESS SOFTWARE**

Professionally designed to utilise the full power of CP/M on the Amstrad CPC464 Disc Drive, the suite of packages include:

**Invoicing, Sales, Purchase and
Nominal Ledgers, Payroll, Stock
Control and the powerful CAMBASE
Database System.**

Single Drive Systems from £39
including VAT & carriage

Send for detailed data sheets to:

Cambrian Software Works
Unit 2, Maenofferen
Blaenau Ffestiniog
Gwynedd

Dealer Enquiries Welcome

Anxious to improve?

A BOOK written to show beginners how to use the full potential of the Amstrad has been published by Micro Press.

Basic Programming on the Amstrad, by software author Wynford James, costs £6.95.

It attempts to build up the Amstrad owner's Basic competence and confidence to the point where structured programs can be written in the simplest and most efficient forms.

The book includes programs for graphics which demonstrate animation, games and a data-base using cassette files.

'Ghost hunters' Amstrad act

IT seems that Amstrad computers – along with ghoulies and ghosties – may be among the things that go bump in the night.

A scientific body which normally investigates strange phenomena ranging from the Loch Ness monster to UFOs has turned its attention to the machines.

Roger Morgan of the Association for the Scientific Study of Anomalous Phenomena (ASSAP) has written to *Computing with the Amstrad* for help with his research.

"Can I appeal to readers for

any information, at first or second hand, no matter how bizarre, concerning unexplainable malfunction or unexpected output?" he asks.

Contacted at his London home, he explained: "We are looking for things like strange messages suddenly appearing on screens".

ASSAP, founded three years ago, has some 300 members across the country who devote much of their spare time to serious investigation of the paranormal and related fields.

It was recently called in to

NOW SMITHS JOIN IN

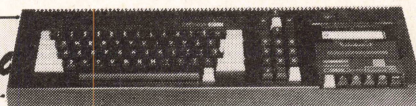
THE Amstrad, its peripherals and software are available at more outlets than ever this year.

W.H. Smiths and Currys have joined Dixons, Rumbelows, Boots, Comet and Menzies as major retailers.

"This is the first time we have stocked any kind of Amstrad

product" said W.H. Smiths merchandise controller John Rowland. "We are very pleased to be handling the CPC 464".

Amstrad's distribution company Europa Electronics has now signed up more than 1,400 independent dealers of which over half are specialist shops.



MINI OFFICE BECOMES AN EDUCATIONAL AID

MINI Office, the chart topping business software package recently released for the Amstrad from Database Software, has been selected as a national teaching aid.

It will now be incorporated as part of a series of special courses held throughout the UK to link education with industry.

The training program is organised by the Careers Research and Advisory Centre (CRAC) for sixth form students, undergraduates and careers and business studies teachers.

CRAC is a registered charity

funded by most of the UK's blue chip companies ranging from the Abbey National through to Marks and Spencer and Williams & Glyn's Bank.

"Our brief is to increase understanding of business enterprise, the role of management and the kind of skills required", says Maureen Curson, CRAC's course manager.

"So we are very interested in Mini Office to help get our message over".

Mini Office is a professionally written suite of four programs — a word processor, spreadsheet,

database and graphics — which converts an Amstrad into an inexpensive office tool.

However it is the revolutionary pricing of the package — just £5.95 — which has guaranteed it being a runaway success. For business software packages often carry price tags of several hundred pounds.

But CRAC has something else in mind for Mini Office apart from being a teaching aid.

"We also intend to use it to help streamline our own office", Maureen Curson told *Computing with the Amstrad*.

Buzzword article causes red faces

A TUTORIAL software guide for the Amstrad CPC464 has caused red faces after it was discovered it contains material. "lifted" almost word for word from *Electron User*, a sister publication to *Computing with the Amstrad*.

This came to light after Mike Cook, the magazine's technical editor and author of the program, spotted "startling similarities" between what he had written and what is now being offered to Amstrad users.

"Imitation may be the most sincere form of flattery, but this was a bit cheeky", he told *Computing with the Amstrad*. "It was a blatant breach of copyright".

The offending item in "Amstrad Basic. A Tutorial Guide" is a buzzword generator, said to be written for the CPC464 by Dave Atherton. He turns out to be the software manager for BBC Publications.

"The whole thing is a 95 per cent direct copy of a program of mine which appeared in *Electron User* under the title: 'Compose your own Buzz Word Generator', "insists Mike Cook.

"Not that the actual concept of a buzzword generator is new. But for him to have so clearly copied it means he hadn't even bothered to really think about it himself.

"He even used most of the same variable names — and these are so idiosyncratic to specific authors that they become like a signature.

"I think — to say the least — that he's been most unethical..."

When confronted with the allegations Dave Atherton said: "Perhaps I did. Is there any copyright on words?"

"I did read it, and I daresay it did stick in my mind while I was writing the program. All I can say is I'm sorry if I've trodden on anyone's toes".

ask: Does the as a medium?

investigate reports of hauntings at Marylebone magistrates court and has developed an infra red video recorder to assist in its work.

Why has ASSAP suddenly become interested in computers?

"We feel they are a valid subject in the light of the fact we have collected some very interesting data from things run on electricity", says Roger Morgan.

As a town planner, he regularly works with a computer and this has led him to believe there is a possibility that the machines may lend themselves to acting as mediums.

Secretary of ASSAP is Dr Hugh Pincott who also believes computers may well act as vehicles for psychic phenomena.

"A particular interest of mine is regressive hypnosis where people reveal what apparently happened to them in past lives", he says.

"Now one of the areas under

investigation is the possibility of a cosmic database — a sort of big computer in the sky.

"Of course there may be nothing in it. But we have had enough reports to suggest that it is a valid subject for scientific research.

"And we believe that somewhat more ordinary computers may fall into the same category".

Is there anyone out there — whether Amstrad user or even the computer itself — who can help? If so please contact Roger Morgan, 15a Kensington Court Gardens, London W8 5QF.

Programs are free

AN outstanding free software offer is being made to purchasers of the Amstrad CPC464. Every new machine is accompanied by a 12-pack of cassette programs worth £108.40.

Titles include Harrier Attack from Durrell, Roland in the Caves, Roland on the Ropes and The Galactic Plague from Indescomp, Oh Mummy and Sultan's Maze from Gem, Fruit Machine by Paul Aitman, and Bridge-It by Epicsoft.

Educational software consists of Hangman, Animal, Vegetable, Mineral and Timeman One from Bourne Educational Software.

The selection also includes the Easi Amsword word processor program.

MATHS FOR MINORS

SPECIALIST education software producer Applied Systems Knowledge is issuing its new mental arithmetic program, Number Painter, for the Amstrad.

It is the first ASK own-label learning program and is aimed at the home use educational sector for children aged five to 14. Price is £8.95.

WE saw last month how to write our own programs, however primitive. Now we'll look at some ways of improving them. I don't guarantee that you'll be able to produce spectacular programs by the end of this article, but you will certainly be well on the way to an understanding of Basic.

First, though, let's recap a little: We saw last month that a Basic program consists of a numbered sequence of instructions to the computer.

To enter one of these instructions we simply type the correct line number, followed by the appropriate Basic keyword, then press Enter.

As we discovered, because of the line number the Amstrad doesn't do what you tell it immediately but remembers it as part of the program.

To see all the instructions in a program we type:

LIST [Enter].

To actually get the Amstrad to carry out the sequence of instructions we type:

RUN [Enter].

To clear a program from memory (and we should do this before entering a new program) we use:

NEW [Enter].

We saw that we tended to enter line numbers in steps of 10 to allow us to fit in other instructions between them if necessary. Also we found that we could replace a line with a better version by simply giving the new version the line number of the old one.

Finally, to delete a line completely we simply type the line number and press Enter.

Program I is the one we started with last month. Before we continue, type it in and run it, to make sure you know what's going on:

```
10 PRINT "PROGRAMMING"
20 PRINT "IS"
30 PRINT "EASY"
```

Program I

Program II is another way of achieving exactly the same output. Type it in and try it.

Apart from its being an incredibly long-winded way of doing things,



```
10 A$="PROGRAMMING"
20 B$="IS"
30 C$="EASY"
40 PRINT A$
50 PRINT B$
60 PRINT C$
```

Program II

what else is going on?

Well, as you will recall from the first article in this series, the words inside quotes are known as strings – because the computer simply remembers them as strings. That is, it considers HAMSTER as H, followed by A, followed by M and so on, with no idea of the word's meaning.

I don't think that it takes all that much imagination to see that when your computer is printing a lot of output you might be using the same string rather a lot.

For example, in a business letter you might use the name of the company fairly frequently – for example, BBC for British Broadcasting Corporation. The Amstrad's Basic allows us to use much the same idea,

but more as labels than abbreviations.

For instance, in line 10 of the above program we have labelled the string "PROGRAMMING" with the label A\$.

In computer terms we have assigned to A\$ the value "PROGRAMMING".

All this means is that from now on wherever I want to use "PROGRAMMING" in my program I can replace it with A\$. So line 40, which is

40 PRINT A\$

causes the micro to print out "PROGRAMMING".

Admittedly in this example this technique of labelling doesn't save much space or effort, but if the program uses the word "PROGRAMMING" 100 times there would be a substantial saving in using A\$ instead of the string itself.

Similarly, line 20 causes B\$ to label "IS" and line 30 labels "EASY" with C\$, so that lines 50 and 60 give the appropriate printout.

Notice the following points:

- We have chosen our labels so that

they consist of a letter of the alphabet followed by the "\$" sign. Actually, we don't have to restrict ourselves to just one letter, as we shall see, but our label must end with the "\$" sign, since this warns the computer that we are labelling a string. (We'll see later how to label other things.)

- While I used A\$ for the first label, B\$ for the second and C\$ for the third, this was totally arbitrary on my part – labels don't have to follow alphabetic or any other kind of order.

- Although we use an equals sign ("=") to connect the label with what it is labelling, it is safer, as we shall see, not to think of it as an equals sign – think in terms of *A\$ becomes "PROGRAMMING"* rather than *A\$ equals "PROGRAMMING"*.

- We must have the label on the left and what is labelled on the right of the equals sign. A line such as:

10 "PROGRAMMING" = A\$

just does not make sense to the CPC464. Try it for yourself!

- When labelling we put the string inside quotes, as we did previously when using the PRINT statement to print out strings. So line 10 reads:

10 A\$ = "PROGRAMMING"

From now on A\$ completely replaces "PROGRAMMING", quotes and all, so that when we say

PRINT A\$

we don't have to use any quotes – they're already there, implicit in the label A\$.

Have a look at Program III. It's virtually identical to Program II except for lines 40 to 60. Here, instead of using A\$, B\$ and C\$, we use the lower cased versions, a\$, b\$ and c\$.

```
10 A$="PROGRAMMING"
20 B$="IS"
30 C$="EASY"
40 PRINT a$
50 PRINT b$
60 PRINT c$
```

Program III

However when you run the program this makes no difference – the output is the same as in Program II. This is rather odd – you have, for instance, given a value to A\$ in line 10, and managed to print it out using a\$, in line 40!

The point is that as far as the

Amstrad is concerned labels that contain the same letters are identical – whether they are in upper or lower case. So:

PRINT A\$

is the same as

PRINT a\$

Beware – not all micros are like this . . .

Now when we label a string the label refers to whatever is inside the quotes, including spaces, as you will see if you run Program IV:

```
10 REM PROGRAM IV
20 MODE 1
30 A$ = "TEST"
40 B$ = " TEST"
50 C$ = "  TEST"
60 D$ = "   TEST"
70 PRINT A$; B$; C$; D$
80 PRINT "0123456789012345678901234567
890123456789"
```

Notice that our punctuation – semicolons and apostrophes – works for labelled strings just as it worked on its own.

Notice also that we have introduced a new Basic keyword in line 10 – REM. We use REM, which is short for REMark, to add comments or

headings to our programs.

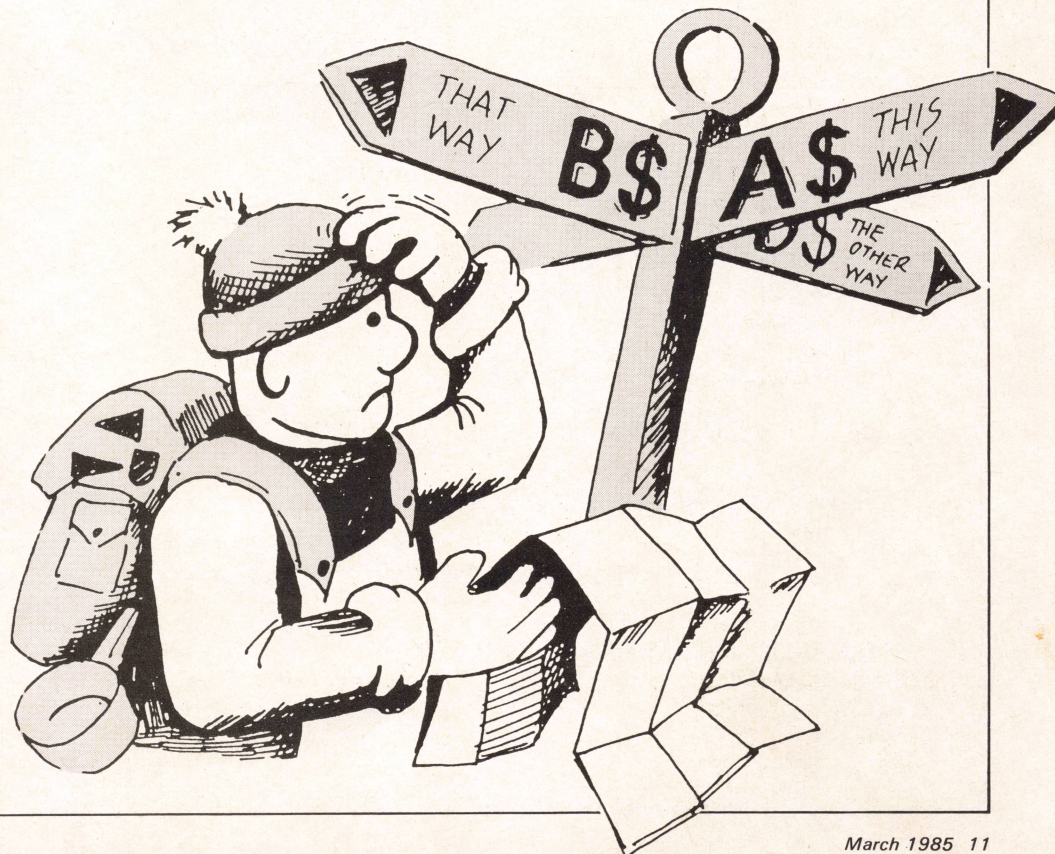
When the Amstrad encounters REM in a line it ignores everything else after it on the same line. This means we can write whatever we want after REM (providing it is on the same line) without fear of the micro giving us an error message – the CPC464 doesn't "read" the line beyond the REM.

If we use REM to prefix our comments on the program we can annotate our program. Certainly each main subdivision should have one or more REM statements explaining what is going on.

Since the Amstrad ignores the contents of REM statements you could leave them out of your program entirely and it will work as effectively. However it is good programming practice to include them.

In the program examples I have used a single REM at the beginning of the program, as it is so short. Bear in mind however, that REM can appear on any line in a program.

Now for some jargon. From now on we shall refer to our labels as variables. Don't be put off by the mathematical sound of that – they are still just labels! And instead of saying we are labelling, we say we are assigning, as we have mentioned



previously. The actual string involved is known as the value of the variable.

So:

A\$ = "TEST"

reads "the string variable A\$ has assigned to it the value 'TEST'". The actual act of giving a variable a value is called an assignment.

To return to the world of actual programs, you can mix and match string variables and actual strings however you want. Program V illustrates the point:

```
10 REM PROGRAM V
20 MODE 1
30 A$ = "MY NAME IS"
40 B$ = " MIKE"
50 PRINT A$; B$
60 PRINT "MY NAME IS";B$
70 PRINT A$;" MIKE"
```

Notice the space of the beginning of the string assigned to B\$ – you need this otherwise the output looks rather odd. Leave it out if you don't believe me!

As we saw last month, a

'you can mix and match string variables however you want'

semi-colon at the end of a line causes the next output to start immediately after the last and not on a new line – as it would do in the absence of the semi-colon. That is, it "glues" the strings together.

The internal semi-colons of lines 50, 60 and 70 do much of the same, "gluing" variables to strings, etc.

While this is grammatically correct Basic, the Amstrad assumes (unless you tell it otherwise) that variables and strings mentioned in the same PRINT statement are meant to be output continuously on the same line. To prove this run Program IV omitting

all the semi-colons.

You've got to be careful here, though. If you typed line 50 as:

50 PRINT A\$B\$

– that is, with no space between the variables – the program would still work. This is because the Amstrad recognises the "\$" as a delimiter of the string.

Be careful of running variables together like this though. It can cause problems later – and it makes your programs very hard to read. Stick to:

50 PRINT A\$ B\$

if you want to do this sort of thing.

Also, while we're on the subject of grammatical propriety, when we're assigning variables we should use the LET statement. So line 40 should read

40 LET B\$ = "MIKE"

As you've already discovered, we can omit LET altogether.

● Next month, more on variables and INPUT – which opens the door to effective programming.

**UPGRADED TO THE CPC464 AND LEFT YOUR HARDWARE BEHIND?
RE-CONNECT TO THE OUTSIDE WORLD WITH THE**

K.D.S

RS-232 AND PARALLEL INTERFACES

RS-232

Independent RX:TX Baud (75-2400)
Handshake Facilities
Printer Controlled from BASIC
RX Data Buffered to Centronics
Integral Converter for
True RS-232 Voltage Levels
Standard 25 'D' Connector

£45.95

Price incl VAT & P/P

PARALLEL

Twin 8 bit Ports
Operates direct from BASIC
4 Programmable Operating Modes
Handshake Facilities
An Ideal Unit for Controlling
8 bit printers or the
Robot in Your Life

£25.95

Both units cased and include through connector for interstacking or connection of further add-ons (disc drive etc.)

Literature supplied and software for RS-232 on tape

PROJECTS UNDER DEVELOPMENT

SPEECH SYNTH
SIDEWAYS ROM
MODEM
LIGHT PEN

TEL (04853) 2076

K.D.S. ELECTRONICS

15 HILL STREET
HUNSTANTON
NORFOLK
PE36 5BS

THERE'S one thing you can say for certain about the new Amstrad disc drives — once you've used them, there's no way you're going to be satisfied going back to using cassette!

Discs score over cassette because of the exceptional increase in speed they give you. Files that would have taken many minutes from tape are loaded and saved in seconds.

And a disc system goes straight to the file you want — there's no waiting for tape to wind round to the right position.

Upgrading from cassette to disc couldn't be easier. The DDI-1, as it is formally known, consists of two parts — the interface unit, which simply plugs into the floppy disc edge connector, and, connected to it by a ribbon cable, the disc drive itself.

The DDI-1 uses three inch discs rather than the more standard five and a quarter. Although I'm more used to the latter on other micros, I was quickly convinced of the advantages of these compact discs, as they're known.

The disc inside may be floppy, but you wouldn't know it. It's enclosed in a rectangular plastic case that thoroughly shields it from the outside world. This makes the discs far more user-friendly than their larger colleagues, which need careful protection from dust, smoke and so on.

Once you've got it all connected up and switched on your Amstrad (after switching on the disc drive, which has its own power supply) you're automatically working in Amsdos, Amstrad's own disc operating system.

From now on any commands that would have been sent to the cassette are sent to the disc drive instead. That is, CAT, LOAD, SAVE, MERGE and so on work as normal, but on the discs.

The disc drive is considered to be stream nine (as was the cassette previously), so commands such as INPUT#9 and PRINT#9 operate on the discs.

This compatibility of commands makes upgrading to discs very simple — Basic filehandling works just as before. (Speed Write, of course, becomes redundant.)

There is a difference in the length of filenames, however. The filename consists of two fields, separated by a dot. The first can be up to eight

Discourse on discs

First in an occasional series by MIKE BIBBY

characters long. The second, which is optional, is called the filetype, and can be up to three characters long.

Incidentally, you must specify a filename. You cannot, as you can on cassette, enter something like LOAD"".

Amsdos can handle the usual types of files — Basic, Ascii, protected and binary. Also if you haven't specified a filetype, Amsdos will supply one for you — BAS for Basic and BIN for binary files.

One nice touch is that if you save a file as, say TEST.BAS, and there's already one by that name on your disc, the first file isn't overwritten, but its filetype is altered so it reads TEST.BAK.

You can also have wildcards in filenames — ? for a single character and * for more than one.

Using Amsdos from within programs is generally quite simple — it's so like the cassette. However I wasn't satisfied with the error messages. A flawed program such as:

```
10 OPENIN "TEST"
20 PRINT #9, A$
30 CLOSEIN
```

should come back with a more meaningful message than "Break in 20".

Also, it would have been nice to have an accessible read/write single byte operation from Basic — particularly as there's no built-in support for random access files.

The weakest aspect of Amsdos however is the external commands — these are contained in the ROM fitted in the interface, and are preceded with the ! symbol (found on the @ key).

Some are quite useful, such as !A and !B which select drives A and B respectively — the DDI-1 comes with a connector fitted to the cable, ready to take a second drive.

!TAPE "switches out" the disc unit and selects cassette operations, while !DISC restores the disc. There are also useful variants of these

created with the suffixes ".in" and ".out".

You could, for instance, load a file from tape and resave it on disc with the sequence:

```
!TAPE.IN
!DISC.OUT
LOAD "TEST.BAS"
SAVE "TEST.BAS"
```

The effect of the first two commands is that the micro will take its input from the tape, and output to disc.

However any DOS (disc operating system) worth its salt should have an easy way of renaming and erasing files. The way Amsdos has chosen to handle these processes is far from easy.

Unless there's something I've been missing, the only way to use Amsdos to give a file called "OLD.BAS" the title "NEW.BAS" is to store the titles in two variables and then use the !REN command, as in:

```
o$ = "OLD.BAS"
n$ = "NEW.BAS"
!REN,o$,n$
```

Using !ERA to erase a file is equally roundabout. The only user-friendly way of erasing and renaming files is to use CP/M — the alternative disc operating system supplied with the DDI-1.

CP/M is one of the most widely used disc operating systems, running on many types of micro.

Now computer knowalls will go on for hours about the drawbacks of CP/M. For all that they still tend to use it — and for good reason. It's the industry standard, and there are literally thousands of CP/M applications packages.

So having CP/M is a decided advantage. The version used on the Amstrad, CPM 2.2, is stored partly in the ROM and mostly in memory.

For the purposes of this review we'll have to confine ourselves to a

brief overview of CP/M. We'll cover it in more detail in a later issue.

CP/M itself has five built-in commands:

DIR Effectively performs a CAT.
ERA Erases files.
REN Renames files.
TYPE Prints a file byte by byte to the screen.
SAVE Performs a rather specialised sort of save of a file.

However the CP/M master disc comes laden with utilities which extend the CP/M commands far beyond this Basic structure.

The first thing you'll want to do is to format a disc. Formatting organises the disc so that the micro can use it sensibly.

It divides it into 360 different areas, 40 circular tracks of nine

sections or sectors each. Each sector consists of 512 bytes. Unless a disc has been formatted, it can't be used.

Since Amsdos doesn't have a format command of its own you have to do this from CP/M. Assuming you've got the CP/M master disc in drive A, to enter CP/M from Amsdos you type:

!CPM

which hands over control from Amsdos to CP/M.

You will then get the standard CP/M prompt:

A>

Briefly, to load files from a CP/M disc you simply type in name of the file and press Enter. To format, enter **FORMAT** and the program of this name will be run – giving you all the prompts you need. In effect **FORMAT** has become an extra CP/M command.

Also provided on the master disc are utilities for transferring files and copying discs, a PIP utility (which controls the streaming of data in and out of the micro) and several other

helpful items.

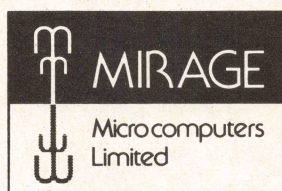
In addition there's the usual CP/M bundle of software – an 8080 assembler and debugger, a context editor for use with the assembler, a hex file dump and batch processing programs.

To get the most out of these you'll need "Soft 159 – A Guide to CP/M". The manual, though adequate, is not overly informative on any aspect of the DDI-1. It does cover the fundamentals of disc operations competently, however.

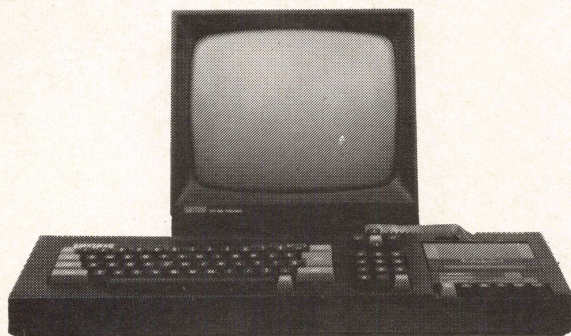
Also included in the package is a full implementation of Logo – a review of which we'll leave for another day.

Overall the system worked well and, minor niggles aside, seems to be well thought out and easy to use.

I shall be very reluctant to return my review drive to our kind friends at Northern Computers. Certainly at £199, the DDI-1 represents exceptional value for money and is well worth considering for your next upgrade.



**for the
AMSTRAD CPC 464**



We stock the full range of AMSTRAD computer products, supported by a wide selection of books and software.

We accept official orders from UK Government and Educational Establishments.

Export enquiries welcome.

Showroom opening hours: MON-SAT 9.00am-5.30pm

Mirage Microcomputers Ltd.

24 Bank Street, Braintree, Essex CM7 7UL

Telephone: Braintree (0376) 48321

HISOFT

presents

FONT 464

for the

AMSTRAD CPC 464

FONT 464 is a font designer and character generator especially developed for the CPC 464 microcomputer.

Design your own character fonts and graphic symbols with this very friendly and powerful package.

FONT 464 allows you to create a new design or amend an existing one using set, reset, invert, reflect, rotate, inverse and even animation! Load and save character sets to/from tape, use the new character(s) from BASIC, design your own animated graphics – all this and more with FONT 464.

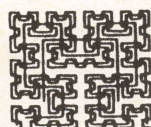
FONT 464 is supplied with three interesting and amusing character sets for you to experiment with.

★ *All this power for: £7.95 inclusive* ★

We also have available for the Amstrad CPC 464:

Hisoft Devpac – our full Z80 assembler and disassembler/debugger with more features than you'll ever need.

Hisoft Pascal – a virtually full implementation of Standard Pascal. Compiles and executes incredibly quickly.



HISOFT

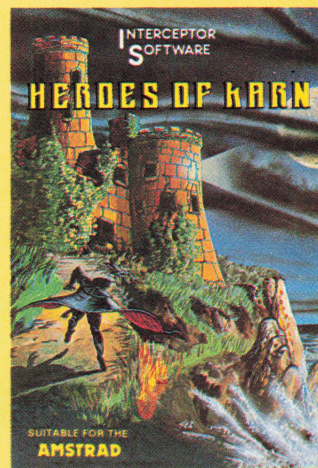
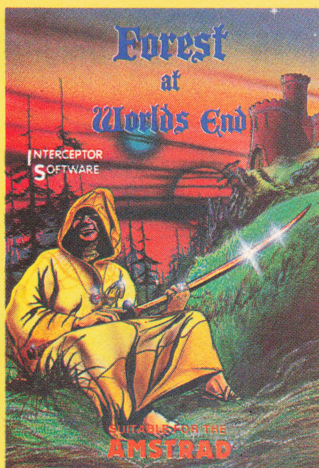


180 High Street North
Dunstable, Beds. LU6 1AT
Tel: (0582) 696421

Amstrad Adventures from Interceptor Software

R.R.P.
£6.00
EACH

Interceptor Software are proud to present their first 4 releases on the Amstrad. These 'user friendly' graphical adventures not only utilise the memory to a full but display the best graphics yet seen on this new computer. A sensible price of £6.00 each makes them the best value software on the Amstrad.

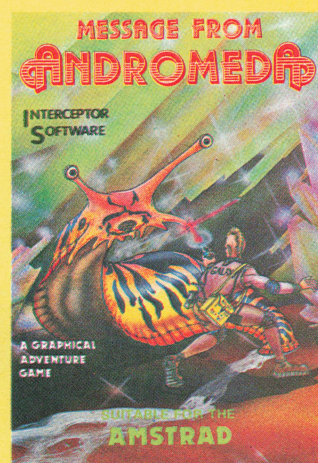
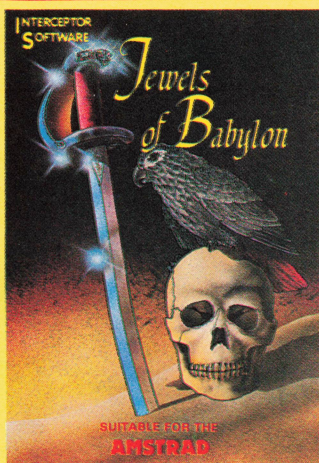


Forest at Worlds End

Forest at Worlds End is a mythical adventure where you have to rescue Princess Mara who has been captured by the evil wizard, Zarn. Many foes await you in the forest!!

Heroes of Karn

Heroes of Karn the hit adventure on the CBM 64 has now been converted to the Amstrad. This classical story will keep you occupied for many moons and when you finally solve the adventure and find the Heroes you will find a vacuum in your life only to be filled with the follow on, Empire of Karn which will be released in 1985.



Jewels of Babylon

Jewels of Babylon is set with a pirate theme, where you, the sole survivor of a pirate raid have the task of recovering the Jewels for Queen Victoria who has promised them as a wedding gift to an Indian Princess

Message From Andromeda

Message From Andromeda is a space adventure where you are a captain of a patrol cruiser in the galaxy. The task set for you is laid fraught with danger? Be prepared for the unexpected.

Available from all leading Software Retailers or direct from

**INTERCEPTOR
SOFTWARE**

Lindon House, The Green, Tadley, Hants, England.

Tel:(07356) 71145/3711 Telex: 849101 INMICS G

It happened one fateful day...

DO you know what an adventure game is? Have you ever played one? If the answer to either of the above questions is "No", then thank your lucky stars. Take my advice – don't ever play an adventure game, not even if it's given to you. Don't even read this article. Stop reading now!

Ah – you seem to have taken as much notice of my advice as the rest of the world does. Be it on your own head...

My wife has always maintained that the micro is the most anti-social creation of man. That is, she did before I discovered adventure games. Now I suppose she'd give them that honour. At least I think so – I never seem to get time to talk to her

nowadays.

But let's go back in time to that fateful day of my first adventure game – Colossal Adventure from Level 9.

As my Amstrad unerringly loaded the program from tape I read the instruction book. I'd chosen a classic to start with. Colossal Adventure is based on Adventure – the original granddaddy of them all.

Apparently I've been given the map to Colossal Cavern, a place of incredible riches and untold dangers.

However as I'm nearing the cavern disaster strikes and I lose it.

I must find the cavern, enter it and return with the treasure, bearing in mind a scrawled note on the margin of the map – *Warning. Magic works in the cavern.*

I was hooked and, looking eagerly up at my monitor, saw the program had loaded. On the screen I saw the message:

You are standing beside a small brick building at the end of a road from the North. A river flows South. To the North is open country and all round is dense forest.

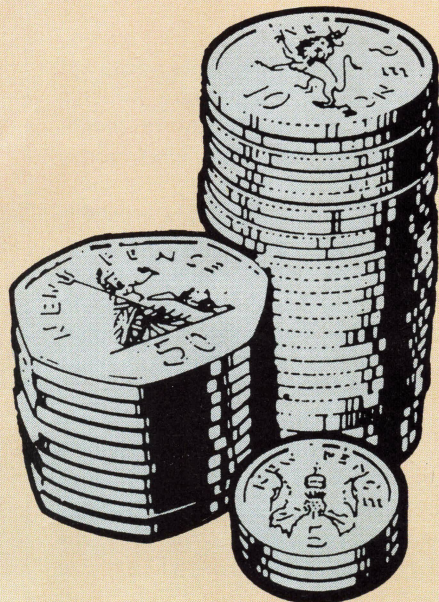
What now?

What now? Never have two words so successfully embodied my feelings. What now? I had no idea. But, when all else fails, read the instructions.

Apparently, what happens in an adventure program is that you wander through an area – be it swamp, forest, cavern or whatnot – meeting dragons, finding treasures, encountering magic and all manner of evil adversaries.

At each stage the computer tells





of keys, a small lamp and an empty bottle.

Now I'm not telling you exactly what I did then, that would be spoiling it. However I can disclose that I managed to get down the well and collect some goodies. I also managed to light my lamp, look round and climb out.

(At this point I can imagine thousands of experienced adventurers yelling, "It's behind you!")

Anyway, I thought I'd be clever. No point in being wasteful by leaving my light on – I might need it later. So I entered:

BLOW OUT LIGHT

and immediately found myself standing outside. It took me a while to figure out why. I suppose you can see it straightaway. As I got further into the game I saw why it was extremely stupid.

Mind you, I thought I was doing pretty well. I followed the river valley to the South, managed to unlock the grate and entered the E-W passage, where I promptly got killed. Three times.

Not that a little thing like that

you where you are, and what you can see.

There are no pictures, just words. But after a while you get used to that. In fact it's like radio – the pictures are much better that way.

The micro then asks you – you've guessed – "What now?"

You then tell the micro – in simple English phrases – what to do. (And believe me I have, many times.)

Among the more successful phrases tend to be simple commands such as MOVE EAST, DROP COIN, TAKE BOTTLE.

TAKE and DROP are very important words. You see, as you move from location to location in the fantasy world you encounter various objects such as keys, coins, black rods and so on.

You'll also be posed various problems to solve before you can move on in the adventure. Often the solutions to these problems depend on the objects you are, or are not, carrying.

Sometimes you won't even be aware that you're being posed a problem – but the outcome can be just as fatal, so learn to be suspicious.

So what did I do? I entered:

INTO BUILDING

and I got in! It would be hard to overstate the sense of satisfaction I felt.

According to the description, the single room consisted of a well with a rusty ladder leading down it into the darkness, and a few objects – a bunch

fact you're down the cavern to your friends and workmates they start showering you with advice.

"Don't forget to make a map".

"When you find an object, wave it – it's always worth a try".

You'll also come across the nervous, defeated type with bags under his eyes and nicotine stains on every visible surface: "Do you know how to get the bird in the cage?"

Don't tell him (just me).

Anyway, after that first trip I was hooked. I'd only got a score of 81 as opposed to the possible 1,100, but I was proud of it, I tell you.

One of the nice things about the game is that you can save where you're up to so you can reload it and carry on another day – a facility which I've made much use of.

And Level 9 offers a free hint service. Just send them the coupon with an SAE and you'll get your one free clue. I haven't used mine yet, but I'm getting close to it!

In fact I'm still nowhere near the 1,100, but I keep on playing. It's that sort of game, combining just the right elements of puzzle, frustration, humour and fascination that the best crossword puzzles and brain-teasers have. And when you've solved the various problems you've been presented with you feel elated, not cheated.

Considering the hours I've played the adventure already, it must be the best value I've ever had in micro games software.

In fact, the only moan I have about it is the horrible gothic typeface the messages appear in. Perhaps I'm getting old, but I found it terribly difficult to read. I would have made a lot faster progress if I'd realised it was a grate I had to open, not a crate!

But that's just a minor niggle. If you're considering buying your first adventure game, or if you're an experienced adventure player who for some bizarre reason has missed out, you should get Colossal Adventure as soon as possible. You won't regret it.



stopped me. After a brief exploration of the hinterland – finding a sheer stone pillar, picnic table and volcano – I was back down that grate armed with fresh knowledge.

All right, I got killed again, but with far more style. And I did work out one or two little tricks – but I wouldn't want to spoil things for you by mentioning them.

Mind you, once you mention the

Paul James



EVENING all, Al here and boy is my face red. I optimistically thought I might have got as far as this issue or even later before dropping the inevitable clanger. But it was not to be.

There it is in the second issue on the first page of my article...

LOCATE (X,Y):PRINT"Hello"

Even the most inexperienced programmer will have realised by now that if you type this in on the Amstrad your micro will respond with "Syntax error". The brackets here are not only not required, they are definitely "anti-Amstrad" and must be omitted.

I'm not making excuses but an error like that when using a word processor rather than including the statement in a program is so easy to make. It's a typing rather than a programming error. The micro of course, would have immediately informed me of the latter.

This is worth bearing in mind by anyone submitting a program to us for publication. Make sure you send it to us on tape or disc, as all our listings are taken from working programs. If as you should, you decide to include an article and list of variables and subroutines with the submission, you must double check the details.

Using a typewriter or word processor for the article, or sending it in hand written, can so easily cause any syntax errors to be missed.

Roland Waddilove's face is also a shade redder as a result of a slight error he made in his game Smiley and the Grumpies in the January issue. We are grateful to Dr Peter Owen of Faringdon, Oxon, for pointing out the mistake. Line 1690 should read:

**1690 score%=score%+bonus%
etc**

He omitted the % from bonus. This doesn't affect the running of the game but it does mean your score will not be as high as it should be because your bonus was not added to it.

There is a lesson to be learned here by the way and it's this. It is so easy to get carried away making sure the graphics of a game work correctly that something fundamental such as



ALAN McLACHLAN prowls the whacky whimsical world of the Amstrad CPC464

a score update can be overlooked. Still Roly will get over it and I hope it hasn't caused any of our readers any heartache.

Right, back to programming. I made myself useful last month by translating Reaction Timer from a BBC Micro program. The more inquisitive of you may be wondering why, at the start of the program, I wasted my time setting the INKs to their default settings.

At the time it was the only solution I could find to a simple problem. I prefer to find my own solutions even if they later turn out to be rather silly. Mind you I had the Great Fire of London down to a faulty 13 amp plug!

If you change the INKs during a program they will remain at their new settings as the program ends.

Consequently if you run the program again it will start it up in the new settings – complete with flashing colours – unless of course like me you are one step ahead of your micro.

I just re-set the colours to their default values to make sure they were as I wanted when the program started. No home micro is going to get the better of me.

Trouble is, no one told me until it was too late that the command CALL &BC02 does it much more simply and with a lot less typing. From now on I will be putting that line in at the start of all my programs.

One of the nicest aspects of publishing a magazine is the enthusiasm of everybody involved. Unfortunately this can lead some people to try too hard.

A prime example of this was the very pleasant, easy on the eye, creamy yellow printed background to Michael Noels' graphics article in the January issue. Everyone in the office particularly enjoyed reading on Page 16:

"For instance, in order for you to see the writing on this page, we've chosen black to be the foreground colour (that is, the colour that letters appear in) and white for the background (the colour of the paper)".

The editor cringes every time he looks at that page, but he's still not in my league yet.

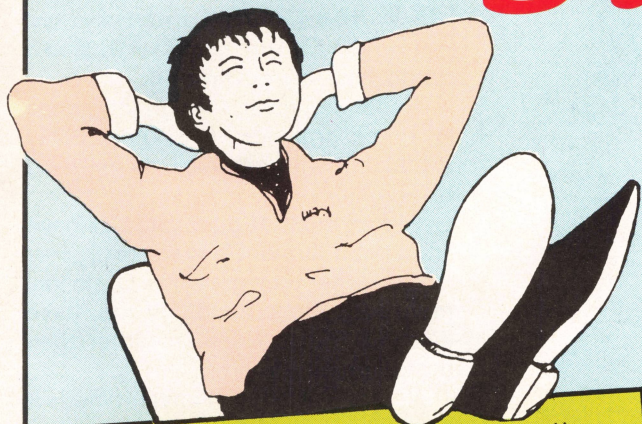
The trouble with him is that he's far too critical. I spent a lot of time one weekend working out the little routine below and when I'd finished – not being an expert programmer – I was delighted with the results.

"Too slow, Al", was the reaction, "much too slow". Well my flabber has never been so gasted and now I'm stuck because I don't know how to speed it up. Any offers?

```
10 MODE 0
12 CALL &BC02
15 RAD
20 ORIGIN 320,200
30 MOVE 0, 100
40 FOR IX= 0 TO 360
50 PLOT SIN(IX)*150,COS(IX)*150,
IX MOD 15 +1
60 PLOT SIN(IX)*75,COS(IX)*150
70 PLOT SIN(IX)*150,COS(IX)*75
80 NEXT
100 FOR IX= 0 TO 15
110 INK IX,0
115 NEXT
120 WHILE 1=1
130 FOR IX= 1 TO 15
140 INK IX,26
150 FOR x=1 TO 100:NEXT
160 INK IX,0
170 NEXT
180 WEND
```

Bye for now.

Our gift to you!



Also packed into this month's cassette and ready for you to run are:

- **STARFLEET:** Earth's defences have been seriously weakened by a shortage of qualified starfleet pilots. Can you come to the rescue?
- **SWAMP:** Hilarious arcade action as you join the Wotawallies in their desperate battle against the giant frogs.
- **PARACHUTE:** Use simple addition to guide your parachute to earth in this educational program for the younger child.
- **DUMP:** An invaluable screen dump utility for Modes 0 and 1.
- **HEXER:** A hexadecimal loader, ideal for the novice machine code programmer.
- **PLUS all the example programs from our Sound, Graphics and Random Reflections articles.**

Still available are the January and February tapes. This is what they contain:

JANUARY

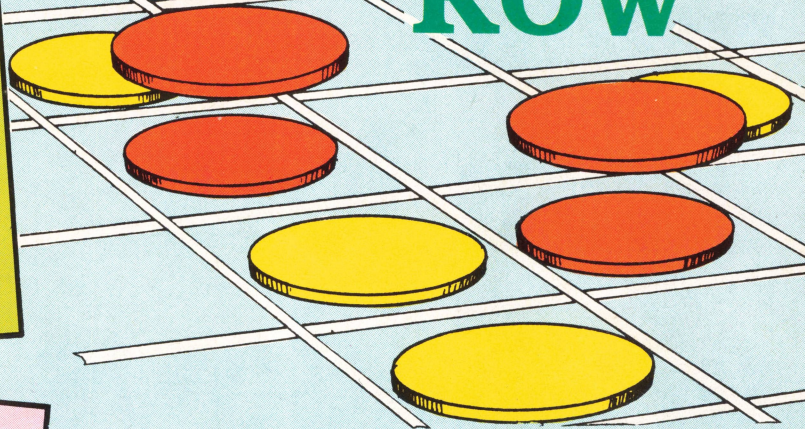
- **SMILEY:** Can you avoid the Grumpies as you guide Smiley round the labyrinth? Guarantees hours of fun.
- **CODE:** You don't have to be a mastermind to play this intriguing logic game – but it helps!
- **BINARY:** Baffled by binary bits? Let our utility help you out.
- **DANCER:** Simple but fun. You'll find our dancer is a lovely little mover.
- **TRAPPER:** You'll need quick-thinking, fast reactions and downright cunning to pen the monster of the maze.
- **SCROLLER:** Add that professional touch to your text with this slick sideways scrolling routine.
- **LETTER LITTER:** Keep your Amstrad tidy and learn the keyboard layout at the same time with this entertaining educational game.
- **PLUS all the example programs from our Sound and Graphics articles.**

FEBRUARY

- **DIGGER:** Addictive arcade action. Can you guide our intrepid hero in his search for crystals and avoid the aliens?
- **SIMON:** Hours of fun with our Amstrad version of this simple children's game. You won't want to stop!
- **KINGDOM OF CRAAL:** A devious fantasy full of surprises, conjured up by the A team. Put your patience to the test and don't take it too lightly! This intriguing adventure will satisfy even the most seasoned adventurer.
- **TEXT EDITOR:** You'll never use a typewriter again once you've used this powerful, yet simple word processor.
- **REACTION TIMER:** Your reactions may not be what they used to be. Test them out on this simple little program.
- **ORIGINS:** Simple but effective, this demonstrates the dramatic effects obtained by changing the graphics origin.
- **PLUS all the example programs from our Sound and Graphics articles.**

There's been such an enthusiastic welcome from our readers for the monthly tape of listings from the magazine that this month we're saying 'thank you' with a very special free gift – a complete full-length game that is far too long to put in the magazine itself. It's called ...

3D FOUR IN A ROW



It's an enthralling three dimensional version of the popular game of luck, skill and strategy. Its nine levels of difficulty make it an entertaining challenge for all tastes and ages.

3D Four In A Row is written to the highest professional standards, with exceptional graphics. And it takes the fullest possible advantage of the Amstrad's advanced design.

ALL this month's programs on one tape PLUS our free gift ... for only
£3.75

HOW TO ORDER

LOCATE that mysterious text screen — and open lots of windows



WE HAVE already looked at all the colours available on the Amstrad and the ways we could fill our pens with them using the *ink* command.

So far, however, though this series is titled graphics, I haven't really explained what graphics, as opposed to text, is.

In fact most of our examples have concerned straightforward text operations rather than graphics. Or, to use more accurate terminology, they have involved the text screen rather than the graphics screen.

You see, although I haven't made it explicit, when you switch on your Amstrad, there are two different "screens" present.

To put it simply, if not entirely accurately, there is the text screen, which you use for writing words on, and the graphics screen on which you draw pictures.

The reason you haven't noticed these two screens so far is that when you switch on or reset, the text and graphics screens overlap — both filling the entire screen.

Let's prove there really are two screens. Reset your micro with Ctrl + Shift + Escape so that the start up message appears. Next enter:

CLS

It won't surprise you that the screen now clears leaving you with the Ready prompt at the top.

Reset the micro once more and this time enter:

CLG

Again the screen clears, but notice — the Ready prompt is now halfway down the screen. Never mind why for the moment — the point is that it shows there's a difference between CLS and CLG.

You see,

CLS

means "clear the text screen" whereas

CLG

means "clear the graphics screen". See, there *are* two screens!

Later on we'll see how we can separate them entirely, or overlap them in various ways by limiting them to screen areas picturesquely called windows.

For the moment, however, we'll leave them both occupying the whole of the screen. To avoid the schizophrenia this may induce we'll consider each screen separately,

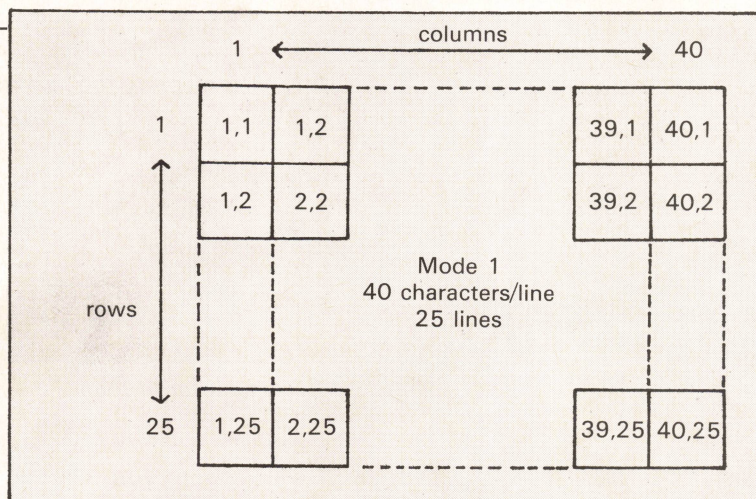


Figure 1: Text screen coordinates, Mode 1

starting this month with the text screen.

The text screen is the screen that our

```
PRINT "Hello"
```

writes characters onto. As we've seen, the size of the text screen – that is, the number of characters we can fit on it – varies from mode to mode.

Each mode allows 25 rows of characters. But Mode 1 allows us 40 characters a row, Mode 0 a measly 20 and Mode 2 a generous 80 characters per row.

We can consider each character to occupy a different "cell" of the screen, each mode giving us a different number of cells.

There's also a coordinate system for these cells, which we've been using to some extent when we've used TAB. The first cell on a row is referred to as number 1, the second is number 2 and so on. Thus:

```
PRINT TAB(5) "*"
```

would put an asterisk in the fifth column, or cell or a row, whereas

```
PRINT TAB(10) "*"
```

would place it in the tenth column.

Program I shows how we can print a simple diagonal of asterisks on the screen using TAB. It's in Mode 1. Try altering line 20 to put the program in Modes 0 and 2, and notice the effects.

The diagonal appears to go further

```
10 REM PROGRAM I
20 MODE 1
30 FOR loop% = 1 TO 20
40 PRINT TAB(loop%) "*"
50 NEXT loop%
```

Program I

across the screen in Mode 0 since there are fewer cells. What cells there are, are larger, however, so the asterisks appear larger.

We say that Mode 0 has coarser resolution than Mode 1. This may seem a drawback, but the advantage is that in this mode you can have 16 colours on screen at a time.

In Mode 2, though, the diagonal fails to travel even halfway across the screen. This is because there are now

By MICHAEL NOELS

80 cells, or columns, in a row. And, of course, the cells are smaller.

Mode 2 has a finer resolution than Mode 1. The drawback is that you can only have two colours on the screen at once.

Experiment with TABbing off the line – for example try:

```
PRINT TAB(44) "*"
```

in Mode 1. As you'll find, the TAB "wraps" round onto the same line to place the asterisk in column 4.

Up to now we've limited our coordinate system to determining the column in a row at which an asterisk appears. By using LOCATE, however, we can have far more control over the position of our asterisk – we can choose both the row and column at which it appears.

Take a look at Figure 1, which shows how this text screen coordinate system works in Mode 1. Each cell has its own pair of coordinates, the row it is on followed by the column it is in – both separated by a comma.

So in Mode 1 the top left hand cell

```
10 REM PROGRAM II
20 MODE 1
30 FOR loop% = 1 TO 20
40 LOCATE loop%, loop%
50 PRINT "*"
60 NEXT loop%
```

Program II

of the text screen would be denoted by 1,1 and the bottom right hand cell by 40,25.

Let's introduce the idea of a text cursor. This is an imaginary sort of marker to show where the next character is going to be printed on the screen. If you enter:

```
PRINT "Hello": PRINT "There"
```

the words will appear on different lines. We say that, after printing *Hello*, the text cursor moves to the beginning of a new row.

If, however, you enter:

```
PRINT "Hello";: PRINT "There"
```

the words will appear "glued" together on the same line. We say that the semi-colon causes the text cursor to stay on the same line.

We could describe TAB(n) as moving the text cursor to column n on the current line.

LOCATE allows us to shift the text cursor and hence the position of the next character printed to wherever we want on the text screen.

We follow LOCATE with the coordinate of the cell we want to place the cursor at. So LOCATE will always be followed by two figures separated by a comma. The first specifies the column and the second the row at which we want the text cursor to appear.

Try the following in Mode 1:

```
CLS: LOCATE 20,5: PRINT "*"
```

As you'll see, the asterisk appears in column 20 of the fifth row. Follow this up with:

```
LOCATE 5,20: PRINT "*"
```

This time your asterisk appears in column 5 of the twentieth row. Make sure you get your numbers the right way round!

Actually here's an easy way to remember which comes first, the C for column comes before R for row in the alphabet – as in CR (Carriage Return).

Try Program II. This should give identical results to Program I. As *loop%* increases, LOCATE moves the

text cursor both to the next row and the next column before printing the asterisk.

In Program I *TAB(loop%)* moved the cursor to the next column as *loop%* increases. Moving to the next row is automatically taken care of by

```
10 REM PROGRAM III
20 MODE 1
30 offset% = 21
40 FOR loop% = 1 TO 20
50 LOCATE loop%, offset% - loop%
60 PRINT "*"
70 NEXT loop%
```

Program III

having no semi-colon at the end of PRINT statement.

By the way, notice that while you can get away with:

```
PRINT TAB(5) "Hello"
```

you can't have:

```
PRINT LOCATE 5,5 "Hello"
```

That is, LOCATE needs to be in its own separate statement. Also, it doesn't have brackets round its numbers, as does TAB.

Since Program II gives identical results to Program I and is one line longer, you may be wondering why we need LOCATE. Well, have a look at Program III.

Being able to specify the row allows you to start at the bottom of the screen and work up, effectively reversing the direction of the diagonal.

We accomplish this in line 50 by making the value of the row selected by LOCATE, *offset% - loop%*.

Initially *offset%* has the value 21, so the first time through the loop, we locate our asterisk at 1,21-1. that is, 1,20, the final character on the twentieth row.

Next time through, *loop%* is 2, so

```
10 REM PROGRAM IV
20 MODE 1
30 offset% = 21
40 FOR loop% = 1 TO 20
50 LOCATE loop%,loop%
60 PRINT "*"
70 LOCATE loop%, offset% - loop%
80 PRINT "*"
90 NEXT loop%
100 WHILE INKEY$="": WEND
```

Program IV

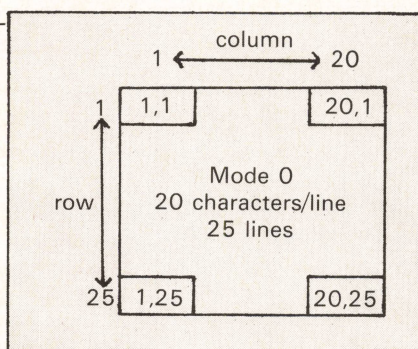


Figure II: Text screen, Mode 0

we locate our asterisk at 2,21-2, = > 2,19 – the second character on the 19th row. Next time through it will be at 3,18 and so on, building up our reverse diagonal.

Program IV is a compilation of Programs II and III that prints a cross of asterisks on the screen. Program V goes even further.

See if you can work out what it does before you run it. A thorough understanding of Program IV will help.

Incidentally, see what happens if the row or column numbers you give LOCATE are too large. For example, in Mode 1 you could try:

```
LOCATE 49,32: PRINT "*"
```

You'll find that specifying an overlarge column number simply places the text cursor in the first column of the next line. Specifying an overlarge row number in effect specifies the bottom of the screen.

The meaning of overlarge depends, of course, on the mode you're in. For instance in Mode 2 specifying a column number 49 is well within scale, since we have 80 characters on a row.

Figures II and III show the sizes of the Mode 0 and Mode 2 screens respectively. Watch out for problems

```
10 REM PROGRAM V
20 MODE 1
30 offset% = 21
40 FOR multiple% = 0 TO 1
50 FOR loop% = 1 TO 20
60 LOCATE multiple%*20+loop%, loop%
70 PRINT "*"
80 LOCATE multiple%*20+loop%, offset% - loop%
90 PRINT "*"
100 NEXT loop%
110 NEXT multiple%
120 WHILE INKEY$="": WEND
```

Program V

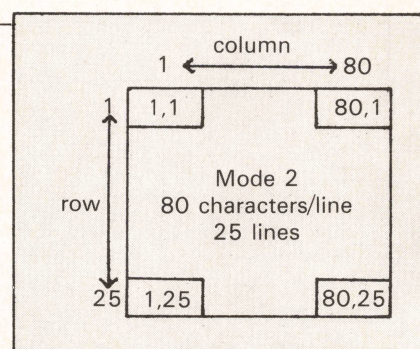


Figure III: Text screen, Mode 2

with LOCATE when you change mode!

To tie all this in with what we've already learned, have a look at

```
10 REM PROGRAM VI
20 MODE 1
30 WHILE -1
40 colour = RND(1) * 3 + 1
50 PAPER colour
60 PRINT " ";
70 WEND
```

Program VI

Program VI. This prints out a series of randomly coloured spaces. One point to note is that a space is a character that's completely background – so to change its colour we use *paper* not *ink*.

Program VII merges these ideas with that of LOCATE, printing random colours at random points on the screen. Notice that, though I'm in Mode 1, I've limited *row* from 1 to 24, and *column* from 1 to 39 – that is, one

```
10 REM PROGRAM VII
20 MODE 1
30 WHILE -1
40 colour = RND(1)*3+1
50 row = RND(1)*24+1
60 column = RND(1)*39+1
70 LOCATE column,row
80 PAPER colour
90 PRINT " ";
100 WEND
```

Program VII

below their maximum value. This is to prevent the screen scrolling.

Pretty, isn't it?

And now for something completely different. Reset your micro (which ensures you're in Mode 1) and enter:

```
WINDOW 10,20,12,16
```

The first thing you'll notice is that the Ready prompt appears in a

peculiar place – towards the middle of the screen. What we've done is to restrict the text screen to a window of the screen – the rectangle shown in Figure IV.

To see this more clearly, enter:

PAPER 3

Notice that your typing doesn't appear "where it should".

To really see the window, enter CLS and the area of the window we've defined will be cleared to red.

Notice also that the command that defined the window is still on the screen – it didn't get cleared. You see, now we've limited the text screen to our window the effect of CLS is also limited to that window.

To prove my point about the text and graphics screens being different, enter CLG and you'll see the whole screen clear this time. After all, we haven't done anything to affect the graphics screen, have we?

WINDOW works like this: The first and second numbers define the left and right hand side of the window respectively, in terms of the text screen's coordinate system. The third and fourth number define the top and bottom of the window.

So, assuming you're still in Mode 1, to restore our window to the full text screen, we would enter:

WINDOW 1,40,1,25

(The 40 would be 20 in Mode 0 and 80 in Mode 2, of course.)

Before you do that try using PRINT, TAB, LOCATE, PAPER and PEN in the window we've created. You'll find that these behave exactly as you'd expect, except for their being on a smaller screen – the text in the window even scrolls when it's full!

Note, though, that LOCATE and TAB are relative to the new window. That is, to place an asterisk in the top lefthand corner of the new window enter:

LOCATE 1,1 : PRINT " *"

You may have thought you need LOCATE 10,12 to do this – after all, that's what you needed when using the old screen.

Once you're in a new window, though, the coordinates for LOCATE "start again" with 1,1 as the top left hand corner, as Figure V shows. Notice, by the way, our window is 5 lines of 11 characters – not 4 x 10 as

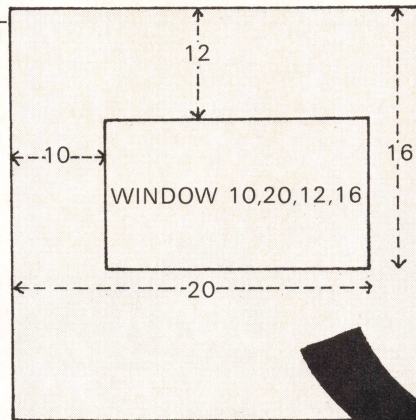


Figure IV: Defining a window

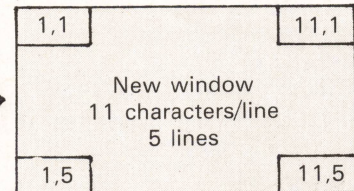


Figure V: Coordinates in new window

the numbers might lead you to expect.

Try experimenting with various windows – you'll soon get the hang of them, and you'll find that they do, indeed, behave exactly as mini text screens.

To get back to the normal Mode 1 text screen you could enter:

WINDOW 1,40,1,25

as described above. A quicker way is to simply change mode.

You may be wondering by now why we would want to use a window. After all, it just restricts the screen, limiting our possibilities. We've already learned how to put text where we want LOCATE on the original screen, so why bother?

Part of the answer is that we can have more than one text window on the screen at once, which allows us to create many special effects.

For instance, the micro might reserve one section of the screen for questions, and print our responses and the correct answer in separate windows.

Often this approach is used in adventure games. One window, occupying the top half of the screen, describes the locations while another, taking up the bottom half, is reserved for your moves, inventories and so on.

We'll be looking at how to use multiple windows in more detail later in the series. This month, though, I'll leave you with an example of using a single window to effect.

One window can be far more effective than you think – in fact, the BBC Micro has only one. For instance, you can use it as a quick way of drawing rectangles. Simply define a window of the appropriate size, set PAPER to the colour you want and

clear the screen. Hey presto – a rectangle!

As we'll see, you could use this technique to construct rough and ready bar charts.

For the moment have a look at Program VIII. Each time you press a key you'll get a randomly defined rectangle (line 50) of a random colour (line 60).

Notice how line 20 sets up the small Enter key to return things to normal once you're out of the program.

See if you can use this window technique to draw three equal squares – colours red, yellow and cyan – in Mode 1.

If you want to investigate the effect of WINDOW further you could try adding a line defining a window to the example programs we've already

```
10 REM PROGRAM VIII
20 KEY 139,"MODE 1: CALL &BC02: PEN 1
: PAPER 0"+CHR$(13)
30 MODE 0
40 WHILE -1
50 WINDOW RND(1)*19+1,RND(1)*19+1,RND
(1)*24+1,RND(1)*24+1
60 PAPER INT(15*RND(1)+1) : CLS
70 WHILE INKEY$="" : WEND
80 WEND
```

Program VIII

covered in the series – remember to put the line *after* the mode change.

That's all for this month. We'll soon be learning lots more about windows, but for the moment I suggest you try consolidating what we've already covered by incorporating the ideas in your own programs. Good luck!

MIKE BIBBY
 continues his series
 of articles aimed at
 lifting the veil of
 mystery cloaking
 the fundamentals
 of the Amstrad
 CPC464's workings

Consider the of nybbles..

AS we have mentioned in previous articles, the Amstrad CPC464 – and all other machines based on the 6502 micro processor – handles its binary numbers in groups of eight bits at a time. Such a group of eight is called a byte.

However, while handling eight bits at a time is satisfactory from the machine's point of view, from the human side of things it's rather difficult to manage. Those 1s and 0s are far too prone to error. Look at Table I for instance. It contains an error – can you find it?

It's all too easy to slip up when handling binary numbers – a single 1 in the wrong place and all is lost! To make things easier to deal with, when I am copying out binary numbers I put a wavy line between bits 3 and 4 to split the byte into two equal groups of four.

For example, if I were copying

% 10001111 (= 143)

I would write

% 1000~1111

Actually, splitting the byte into two groups of four bits is standard practice – each group of four bits is called a "nybble", would you believe.

It's not too hard to see that the biggest number you can represent in a nybble is 15, and the smallest is 0,

%1111 and % 0000

respectively. After all, you've only got four bits to play with!

So we can split up our byte into two nybbles of four bits each. Now when we split up a binary number in this manner we call the "left-hand" nybble the most significant nybble (MSN) and the "right hand nybble" the least significant nibble (LSN).

We have already created one new number system – the binary system. Let's design another one that combines the advantages of the denary system with those of the binary. That is, it will be easy to read and write, yet will still allow us to perceive the binary manner in which the machine handles things.

The system we want is called hexadecimal. This consists of using our standard digits 0 to 9 for the number zero to nine respectively, and the letter A to F for the numbers 10 to 15. In this way it allows us to code the numbers available in a nybble (that is, 0 to 15) with just one digit.

This digit will be in the range 0 to 9 or A to F.

It may take a while to adjust to the idea of using letters of the alphabet for numbers, but it soon becomes second nature.

You just have to get used to counting:

1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

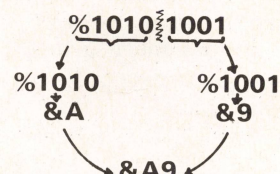
Remember, there are B people in a cricket team, D in a rugby league team and F in a rugby union team. There are C months in a year, and E

days in a fortnight.

Now just as we prefix all our binary numbers with %, we prefix our hexadecimal numbers with &, to avoid confusion. So &F means 15, while &9 means 9.

Studying Table II will really pay dividends – I suggest you practice writing down bit patterns of nybbles and their hexadecimal equivalents until it becomes second nature.

Given that we can encode a nibble in one hexadecimal digit, and that a byte consists of two nybbles, it should readily be apparent that we can encode a byte as two hexadecimal digits side by side, for example:

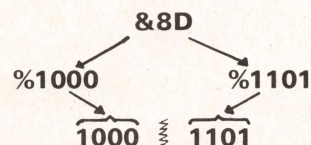


That is:

%10101001 = &A9 = 169

You just split the byte up into two nybbles – a left hand and a right hand nybble, encode each as a hexadecimal number, then put the two side by side.

You can go from hexadecimal to binary just as easily:



e significance

That is:

$$\&8D = \% 10001101 = 141$$

Although you have probably never thought of it in these terms, you are well aware that the value a digit represents depends on the column it is in. The number 230 is not as large as 320, though both numbers contain the same digits.

In hexadecimal coding too the

%10111011 = 187
%10101101 = 173
%10001111 = 151
%11110110 = 246

Table I

column a digit is in is important. For example, &10 is far greater than &01. In binary each column is worth twice the preceding one. In denary, our usual number system, each column is worth 10 times the preceding one. In hexadecimal, each column is worth 16 times the preceding one.

Believe or not, the columns in a four digit hexadecimal number, from greatest to least, are worth:

4096, 256, 16 and 1

respectively. This means that:

$$\& 1101 = 4096 + 256 + 1 = 4353$$

For the moment let's concentrate on the two digit, that is, two column, hexadecimal number, as these are all we need to store our bytes in. In this case the left-hand column is the "sixteens" column, the right hand the units column.

So:
 $16 \ 1$
 $\& 2 \ 1 = 2 \times 16 + 1 = 33$
 $16 \ 1$
 $\& 2 \ D = 2 \times 16 + 13 = 45$
 $16 \ 1$
 $\& 8 \ 0 = 8 \times 16 + 0 = 128$
 $16 \ 1$
 $\& C \ 0 = 12 \times 16 + 0 = 192$

To translate a two digit hexadecimal number into denary simply multiply the number in the left-hand column by 16 and add it to the number in the right-hand column – remembering to translate A to F if necessary.

The second column has the value 16 since the first column can only handle numbers up to 15 (&F) – the largest you can fit into a nybble (%1111). After 15, you *have* to use a second column for 16, that is &10.

Just as in denary, we "carry" at 10 since the largest value our columns can handle is 9, so in hexadecimal we carry at 16, since the largest value our columns can handle is 15 (&F).

It is the fact that we carry at 16 that gives this number system its name "hexadecimal" – here "hex" stands for 6, "decimal" for ten. "Hexadecimal" = 6 + 10 = 16.

Given a second column, &10, as we have seen is 16, 17 will be &11, while &12 is 18 and so on until we reach 31, which is &1F.

We have then run out of legal digits for the units column, so if we want to go on to 32 we had better give ourselves another 16, and set the units column back to zero, that is &20.

Another way of looking at the second column is that it comes from the most significant nibble. To turn the least significant nibble into the most significant nibble, we have to shift it over to the left four times.

If you cast your mind back to last month, this is equivalent to multiplying it by two four times in succession, that is $2 \times 2 \times 2 \times 2 = 16$. This is why a hexadecimal digit representing the most significant nibble is 16 times larger than the same digit representing the least significant nibble.

The largest number you can store in a two-digit hexadecimal number is &FF = $15 \times 16 + 15 = 255$. This is, of

course, the same as the largest number we could store in a binary byte – we often refer to a two digit hexadecimal number simply as a byte.

To obtain the hexadecimal equivalent of a positive interger (whole number) less than 256, we divide it by 16. The quotient is the left hand digit, the remainder the right hand, translating into A to F where necessary.

For example:

$$174 \div 16 = 10 \text{ R } 14$$

That is:

&A R &E

Hence:

$$174 = \&AE$$

Fortunately we don't have to go to such lengths, the Amstrad allows us to simply print out the hexadecimal equivalent of decimal numbers and vice versa.

For instance: **PRINT &BC** will give

188

while **PRINT HEX\$(141)** will give **8D**

Practise your hex for next month!

Decimal	Binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Table II

HAVE you ever been slaving over a program and realised that you have already written a part of it last week? You might have created a space invaders game with a high score table and now you find you're working on a maze game which will also have a high score table.

Wouldn't it be nice if you could borrow the lines from the previous program? And it would save a lot of typing if you could just load them into memory from the copy you saved of the earlier game.

The trouble is that if you have one program (the one you're working on) in memory and try to LOAD another the fickle Amstrad gets rid of the first one and puts the second into its memory. To use a technical term, you're goosed.

However you don't have to be. One of the nice things about the CPC464 is that you can get round this problem by using the MERGE command. This will take a program or subroutine that you've previously saved on tape and add it to the program in memory.

When programs are MERGEed the original one is still there with the one from the cassette tacked onto it.

While seemingly magic, MERGE is a very easy command to use, provided that you follow a few commonsense rules. All it needs is a little practical experience.

For the rest of the article I'm assuming that you've got a tape in your cassette and that you'll type in and save the listings that I ask you to, for later use.

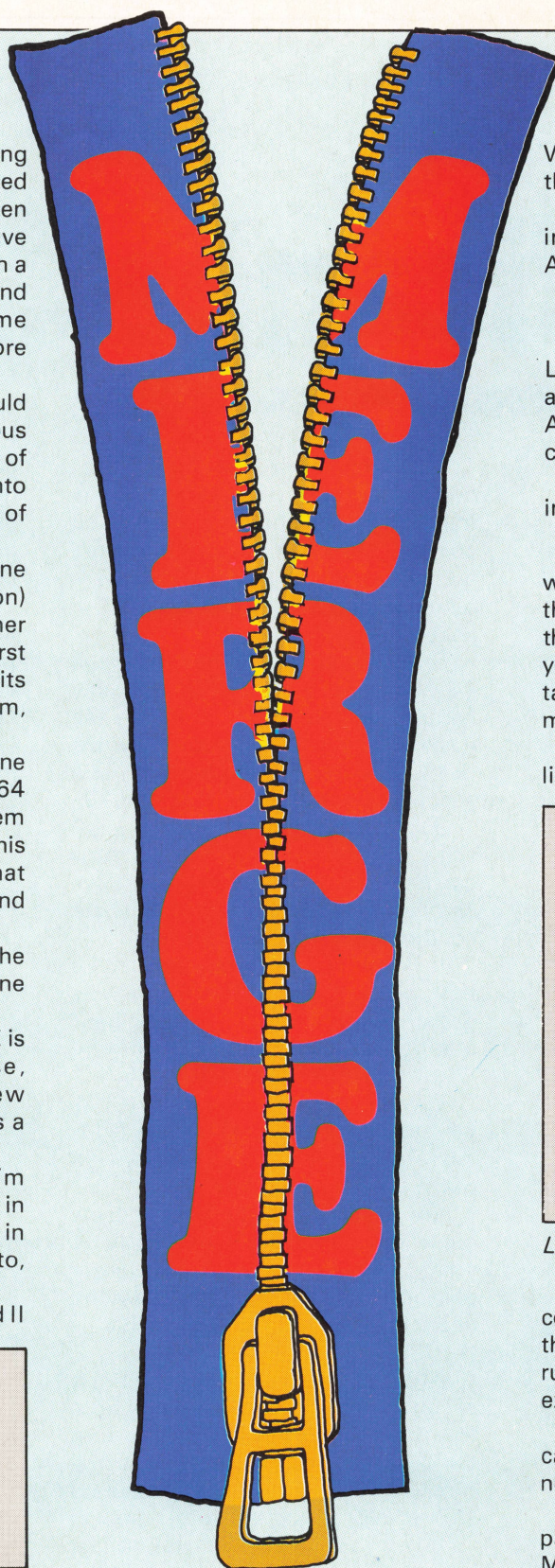
So type in and save Listings I and II

```
10 REM LISTING I
15 CLS
20 PRINT "THIS DOESN'T DO MUCH"
30 PRINT "IT JUST CALLS A"
40 PRINT "SUBROUTINE"
50 GOSUB 200
60 END
```

Listing I

```
200 REM LISTING II
210 REM SUBROUTINE
220 PRINT "THIS SUBROUTINE"
230 PRINT "DOESN'T DO MUCH"
240 PRINT "EITHER"
250 RETURN
```

Listing II



and then enter NEW to clear the memory. Having done that, LOAD Listing I back into the micro.

Now suppose you were engaged in writing a program and you reached line 50 of Listing I. You realise that you already have the subroutine that you want (in our case it's Listing II) buried away on one of your cassettes.

Why not use that rather than do all the typing?

All you have to do is to put the tape in the cassette deck and tell the Amstrad to get on with it using:

MERGE "LIST2"

As you can see, when I saved Listing II I called it LIST2 and now I'm adding it to the program already in the Amstrad. The result should be a combination of both.

You don't have to put a name in the inverted commas.

MERGE ""

will work, but in this case it just takes the first file it finds and adds that to the program in memory. Try it out for yourself, MERGEing Listing II (on the tape) with Listing I (in the micro's memory).

You should end up with something like Listing III:

```
10 REM LISTING I
15 CLS
20 PRINT "THIS DOESN'T DO MUCH"
30 PRINT "IT JUST CALLS A"
40 PRINT "SUBROUTINE"
50 GOSUB 200
60 END
200 REM LISTING II
210 REM SUBROUTINE
220 PRINT "THIS SUBROUTINE"
230 PRINT "DOESN'T DO MUCH"
240 PRINT "EITHER"
250 RETURN
```

Listing III

As you can see, the MERGE command has tagged Listing II onto the bottom of Listing I. You can now run the whole program and read the exciting messages.

If you can't bear to part with it you can SAVE Listing III, but you won't need it again in this article.

So now you know how to add one part of a program to another using the MERGE command. Simple isn't it?

Well, nothing in life is that simple and there are a few points to bear in mind when you've merged the programs.

Suppose you've merged a subroutine with a program you're writing, more or less as we did with Listings I and II. Unless you always use the same variable names when you write programs the subroutine you've

Save time and trouble by using your micro's MERGE command. TREVOR ROBERTS explains in detail

added may not work, or it may work, but badly.

The program you were writing might keep a running total in the variable *total*. The subroutine that you've tacked on might work out an average from a running total but it might hold this running total in the variable *runningtotal*. As you can see there'll be a clash.

This isn't all that much of a problem, provided it's kept in mind. If you've MERGED a subroutine with a program and you get weird results, check that the variables in both parts of the program match.

It helps if, when writing subroutines, you use REMs to make a note of variables used in that subroutine which have to be set up in the main program.

The next point to bear in mind is that when you merge two bits of code the line numbers mustn't clash. Looking at the first two listings, the more suspicious of you may have noticed that they fitted together quite neatly. What, you may ask, if they have line numbers in common?

It's a good question, and to help answer it type in and SAVE Listing IV and Listing V. Enter NEW to clear the micro's memory (I don't want you thinking that I'm cheating), then LOAD and run Listing IV. Exciting isn't it?

When you can tear yourself away, MERGE Listing V (still on tape) with Listing IV (in memory). If you LIST the

```
10 REM LISTING IV
20 FOR X=1 TO 10
30 PRINT "HELLO"
40 NEXT X
```

Listing IV

```
10 REM LISTING V
15 FOR Y=1 TO 10
20 PRINT "GOODBYE"
25 NEXT Y
```

Listing V

```
10 REM LISTING V
15 FOR Y=1 TO 10
20 PRINT "GOODBYE"
25 NEXT Y
30 PRINT "HELLO"
40 NEXT X
```

Listing VI

program now in memory you should end up with listing VI.

The trouble is that if you run it you get GOODBYE 10 times, HELLO once and then an error message.

If you look at the program closely you'll see that although we merged Listing IV and Listing V, some of the lines are missing. Close scrutiny will show that Listing V has survived intact, but the first couple of lines of Listing IV have gone.

The reason is that the listings share line numbers. Listing IV has a line 20, as does Listing V. Obviously you can't have two lines with the same number in the same program, so when you MERGE the listings one of them has to go.

What happens is that when the program in memory and the one being MERGED from tape have line numbers in common, the incoming lines overwrite the old ones. This is how we got Listing VI. Both listings had lines 10 and 20. When Listing V was read in from tape the lines 10 and 20 of Listing IV were overwritten.

This is something to watch out for when merging programs. If line numbers are shared, you'll lose some of them.

The obvious course of action is to renumber either program so that the clash never takes place. Obviously, it's easier to renumber the program already in memory.

Let's try MERGEing Listing IV and V properly. First of all LOAD Listing V from tape. Now use:

```
RENUM 50
```

to renumber it. You'll see that it now looks like Listing VII. Next MERGE

```
50 REM LISTING V
60 FOR Y=1 TO 10
70 PRINT "GOODBYE"
80 NEXT Y
```

Listing VII

Listing IV with this using:

```
MERGE "LIST4"
```

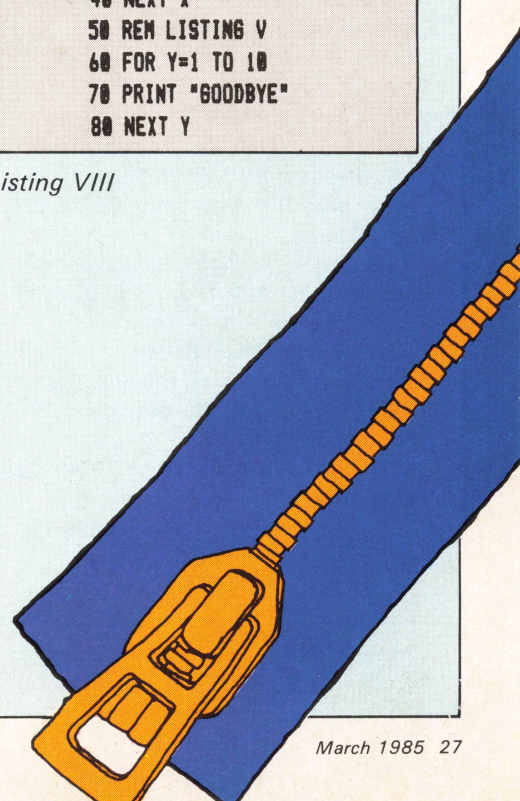
or MERGE with whatever name you've used. The lines no longer clash, and the result is shown in Listing VIII. The programs have been successfully merged.

So, as you can see, taking a program or subroutine from tape and adding it to one in memory is easy. All you have to avoid is overwriting any lines and making sure that the variables are the same in both parts of the resulting program.

The benefit is that, with one simple command, you can save yourself a lot of typing!

```
10 REM LISTING IV
20 FOR X=1 TO 10
30 PRINT "HELLO"
40 NEXT X
50 REM LISTING V
60 FOR Y=1 TO 10
70 PRINT "GOODBYE"
80 NEXT Y
```

Listing VIII



What the hex is it all about?



Part Three of MIKE BIBBY's introduction to machine code

LAST month we looked at hexadecimal numbers – hex for short – and ran two very simple machine code programs. When counting in hexadecimal we went 1,2,3, . . . 9,A,B . . . F.

To avoid confusion with our more usual denary numbers, we prefix hex numbers with the & sign, so strictly we should talk of &F.

If we wanted to count one more than &F we carried one to the 16s column. That is:

$$\&F + 1 = \&10$$

This 16s column meant that:

$$\&23 = 2 \times 16 + 3 = 35$$

and

$$\&AE = 10 \times 16 + 14 = 174$$

With the 16s column we can write the value of any byte. The maximum value a byte can store is 255, &FF in hexadecimal. Once our 16s column was full, we carried one to a 256s column, and after that a 4096s column.

We found that a four column (or digit) hex number was all that we needed to specify one of the Z80's memory addresses. For example, the highest memory location possible, 65535, would be &FFFF.

We split these four digit addresses into two sets of two hex digits – that is, into two bytes. The "larger" pair (the 4096s and 256s columns) formed the hi byte and the "smaller" pair (the 16s and 1s columns) formed the lo byte. For example, &CDEF has hi byte &CD and lo byte &EF.

Even if hex numbering doesn't come naturally to you as yet, I'm sure you've got enough of a feel by now to follow the rest of this series. If you fancy revising hex though, have a look at this month's Bits and Bytes on Page 24.

The machine code routines we covered last month were simple enough. Here's one of them:

CD DB BB C9

Even if you followed the last article closely, the above routine probably doesn't mean too much to you.

Let's have a look at it with its mnemonics:

CD DB BB	CALL &BDBB
C9	RET

Makes much more sense, doesn't it? Particularly when I tell you that the

machine code routine at &BDBB clears the screen.

Remember, though we write the machine code the Z80 handles in hex, it's much easier for us to follow the more readable mnemonics. Using mnemonics this way is known as using Assembler or Assembly language.

Actually, the call to &BDBB clears the current graphics window, but I'm assuming you've just switched on, so the whole screen goes, as no special window is defined.

You'll probably recall that CALL (opcode &CD) tells the micro to "gosub" to an address specified by the following two bytes.

Notice that though the address of the routine is &BDBB the bytes appear lo byte first – that is &CD followed by &BB. This is standard for the Z80.

There's a text screen equivalent of this routine at &BB6C. It clears the text window which, since we haven't set any, in effect clears the whole screen.

If we were going to use this routine instead of the first, our code would now read:

CD 6C BB C9

In assembler this would be:

CD 6C BB	CALL &BB6C
C9	RET

To put this routine into your micro at memory location &3000 onwards use the following:

POKE &3000,	&CD
POKE &3001,	&6C
POKE &3002,	&BB
POKE &3003,	&C9

To run it, we enter:

CALL &3000

The screen clears, and to prove that this time it's the text window we've cleared, the "Ready" prompt appears at the top of the screen.

Remember, when we issue the Basic command CALL &3000 we tell the micro to start performing a machine code routine which has been stored in the sequence of bytes starting at &3000.

The first instruction of this routine (&CD) tells the micro to "gosub" to another routine which is located at &BB6C. The micro knows we mean this address because it's stored in the two bytes directly following the &CD – lo byte first.

When it returns from this routine,

which is already written in the firmware, it looks at the next byte after the two specifying the address to find its next instruction.

In the above program this is &C9, which means RETurn to where you came from – in this case Basic.

All this poking bytes into memory one by one is a rather messy business, though. Surely there's an easier way to enter our hexadecimal numbers?

Program 1, called Hexer, is the answer. It is what is known as a hexadecimal loader, storing the hex numbers you type in into memory.

However, it's much more than that, as you'll see. In fact it's vital that you have this program on tape if you're going to follow this series.

I suggest you type it in very carefully and save it before you try running it. Even if it's entered correctly, you might wipe it out by careless testing – so save it first!

Of course you could save yourself the bother by getting this month's tape – it's on there along with all the other programs from this month's issue. (See Page 55.)

When you run it you'll see displayed:

You may:

1. Enter code
2. Examine code
3. Alter code
4. Run code
5. End program.

To choose any of these options you simply press the number corresponding to it. Option 5 should be self-explanatory. As the reason we wrote it was to enter code, choose option 1. You'll see displayed:

Start address?

This is asking where you want your machine code program to start from – that is where you want the sequence of bytes to begin.

Now this program, for reasons we'll explain later, is tailored to start machine code programs at &3000 for preference.

You can, of course, override its wishes by entering the address you wish to start at. For the moment though simply press Enter and Hexer will default to an address of &3000.

Next, you'll be prompted:

byte?

Here the program is asking you what hexadecimal number you want

```

10 REM      Hexer
20 REM      Mike Bibby
30 REM      (c) Computing
40 REM      with the Amstrad
50 CLEAR
60 ON ERROR GOTO 10
70 size = HIMEM
80 MEMORY &2FF8
90 WHILE -1
100 PRINT: PRINT "You may:-":PRINT
110 PRINT "1. Enter code"
120 PRINT "2. Examine code"
130 PRINT "3. Alter code"
140 PRINT "4. Run code"
150 PRINT "5. End program":PRINT
160 b$ = INKEY$:IF b$="" GOTO 160
170 IF INSTR("12345",b$)=0 GOTO 160
180 b = VAL(b$) : ON b GOSUB 210,400,
550,350,200
190 WEND
200 END
210 INPUT "Start Address"; start$
220 IF start$="" THEN start$="3000":P
RINT:PRINT"Start address = &3000"
230 start = VAL( "&" + start$ )
240 code$="":PRINT
250 WHILE code$ <> "S" AND code$ <> "s"
"
260 INPUT "byte"; code$
270 IF code$="" THEN PRINT CHR$(11);:
GOTO 260
280 IF code$="S" OR code$="s" THEN GO
TO 330
290 code$ = "&" + code$
300 code = VAL(code$)
310 POKE start, code
320 start = start + 1
330 WEND
340 RETURN
350 INPUT "Start Address"; start$
360 IF start$ = "" THEN start$ = "300
0"
370 start = VAL( "&" + start$ )
380 CALL start
390 RETURN
400 INPUT "Start Address"; start$
410 IF start$ = "" THEN start$ = "2FF
8"
420 start = VAL( "&" + start$ )
430 A$ = CHR$(32): PRINT
440 WHILE A$<>"s" AND A$<>"S"
450 PRINT RIGHT$( " " + HEX$(start),4
);" ";
460 FOR loop = 0 TO 7
470 code$=" " + HEX$( PEEK( start+lo
op) )
480 PRINT RIGHT$(code$,3);
490 NEXT
500 PRINT
510 A$=INKEY$ : IF A$="" GOTO 510
520 start = start + 8
530 WEND
540 RETURN
550 INPUT "Alter from"; start$
560 IF start$ = "" THEN start$ = "300
0"
570 start = VAL("&" + start$)
580 code$ = "": PRINT
590 WHILE code$<>"S" AND code$<>"s"
600 PRINT RIGHT$( " " + HEX$(start),
4);" ";RIGHT$( " " + HEX$(PEEK(start)),
2);" ";
610 INPUT code$
620 IF code$="S" OR code$="s" THEN GO
TO 670
630 IF code$ = "" THEN GOTO 660
640 code$ = "&" + code$
650 POKE start,VAL(code$)
660 start = start + 1
670 WEND
680 RETURN

```

Program 1: Hexer

storing in memory, starting at the address you've just selected.

To obtain the previous program that cleared the text screen we enter the following sequence of bytes:

CD 6C BB C9

So in response to *byte* type in:

CD

and press Enter. Notice, you don't need the & in front of CD – Hexer does it for you. &CD will now have been stored at &3000.

You'll immediately be prompted

for another byte:

byte?

Enter 6C, which will be stored in &3001. After the next two prompts enter BB followed by C9. These bytes will be stored at &3002 and &3003 respectively.

Our machine code program will now have been stored in memory, starting at &3000. However you'll still be left with the:

byte?

prompt. Simply enter S for stop (or

Sease, as our illiterate editor would have it) and you'll return to the options menu.

We've now entered our code into memory, but don't be tempted to go straight to option 4 to run it.

As you'll discover all too soon, simple typing errors can play havoc with machine code programs.

I haven't protected the program against silly mistakes such as entering XX as a byte. As I mentioned last month, check and recheck before pressing Enter – if you do you'll avoid making such daft mistakes.

However you might accidentally misread E8 for F8, say, so you need some way to examine memory to see if your code is in correctly.

Option 2 allows you to do this. Select it and, when you're asked for the start address, press enter once more to select the default value. You'll see displayed something like:

```
2FF8 31 0 34 8 0 0 0 0
```

This option displays a sequence of bytes in memory a row at a time, eight bytes to the row. The four digit hex number on the left is the memory address of the first byte in that row.

So in the above case we know that &2FF8 has &31 stored in it (notice the numbers are in hex).

&2FF9 has 0 stored in it, &2FFA (remember we count 8,9,A!) has &34 in it, &2FFB has &8 and so on to the last figure of the row, which is the contents of memory location &2FFF.

This time we've defaulted to &2FF8 instead of &3000 – and it's &3000 we're interested in looking at, since that's where our program is stored.

Well so far we've displayed the row of eight bytes from &2FF0 to &2FFF. Actually, the next byte along would be &3000 since:

```

&2FFF +
  1
-----
&3000

```

To prove it, press the space bar. If all is well you should see displayed something like:

```
3000 CD 6C BB C9 0 0 0 0
```

with the first four bytes holding the routine we've entered.

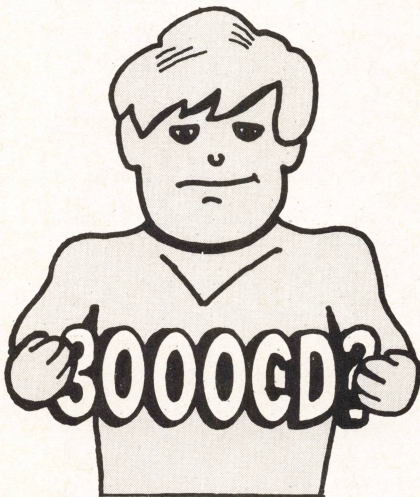
Check carefully that all is well with those first four bytes – they're our

machine code program. If these aren't what you intended to enter you'll have to leave this option (by typing S once more) then re-enter the code via option 1. It's only four bytes, so it's no great hardship. Don't do that yet though.

You may be wondering why I chose to start the display at &2FF8 when it's really &3000 onwards we want, to show our programs. Well I'm going to use &2FF8 onwards as a sort of showcase where I can display the result of the programs resident in &3000 on – so this is quite convenient.

If this bothers you simply enter 3000 when you're prompted for the start address. Notice that, as with all Hexer inputs, you don't need to specify hexadecimal with "&". Hexer expects hex!

Let's assume that our code is now properly entered and checked. We



should have returned from displaying memory to our main menu by entering S. (Alternatively, we could have kept on displaying successive sets of 8 bytes by repeatedly pressing space.)

To run the code we choose option 4. Again, you'll be prompted for a start address. If you simply press Enter you'll default to &3000, our preferred start.

Do so now and you'll see the text screen clear.

Running a machine code program should only be done as a last resort. Unless you're 100 per cent sure that you've got it right don't try it – you're liable to wipe out program, Hexer and all with just a little error.

So if you've selected option 4 to run the code, and decide to back out when you get the start address prompt, simply enter X for eXit and you'll get back to the main menu without running anything.

The same holds for any of hexer's address prompts – X for eXit will take you back to the main menu. Similarly, once you're in options 1 to 3, S for Stop will return you to the menu.

If you've been following, you should have the following program in memory, starting at &3000:

CD 6C BB C9

This isn't too much different from our program to clear the graphics screen, which, you may recall, was:

CD DB BB C9

In fact we've only to alter one byte – the low byte of the address called.

From the main menu choose option 3. You will receive the prompt "Alter from?" Once more you can enter your own start address. In this case though, simply press Enter and it will default to &3000.

You'll see displayed:

3000 CD ?

This tells you that the contents of &3000 are &CD (the opcode for CALL). We don't want to alter this, so just press Enter and it will stay the same. We're now at the next byte in memory and will see displayed:

3001 6C ?

This is the byte we wish to alter, so type in its new value, DB then press Enter. You'll then move onto the next location:

3002 BB ?

We don't want to alter this, or any subsequent bytes, so enter S and return to the main menu.

Points to note about the alter memory option:

- To leave a byte unchanged simply press Enter.
- To alter a byte, type in its new hex value.
- To quit the option type S.

Having altered our code, check it with option 2 (examine code) to ensure we've got the correct code in memory. Finally, run it with option 4.

If you cast your mind back to the first article in the series, we discussed how machine code programs were really about shifting data bytes around in memory.

We likened it to the movement of

trains between railway stations, and pointed out that, just as railway networks have junctions, so in a sense does our memory map of locations.

Instead of moving bytes of information directly from memory location to memory location we use these junctions or registers as halfway houses. We move data from memory to these registers, then from these registers to the new memory locations.

One of the most frequently used, and most versatile registers, is the A register, or accumulator as it is also known.

As with other memory locations it can hold one byte of information. The difference is that the A register is actually inside the Z80, so the microprocessor can get at – or process – the information in the A register in its own special ways.

In other words, there are things you can do to a data byte when it's inside A that you can't do while it's out in the sticks in memory.

So a lot of our programs will tend to involve bringing data into A and similar registers, working on it, then shunting it out.

For the moment though, we won't be doing anything so sophisticated, we're just going to try one or two simple tricks with A.

The first involves a firmware routine hidden at &BB5A. This prints on the screen the Ascii character whose code is stored in A.

By calling this routine we can print out any Ascii character we want on the screen – provided we've managed to get its code into the A register.

Actually, putting a number into the A register isn't that hard. Its mnemonic is:

LD A,n

where LD stands for Load and n is the hex number you're loading into A. So:

LD A,&2A

means Load the A register with the hex value 2A.

Now the opcode for this is &3E. You need to follow it immediately with the hex byte you wish to load into it. So:

LD A, &2A

would translate as

3E 2A

42(&2A) happens to be the code

for an asterisk, so to put an asterisk on the screen we need to:

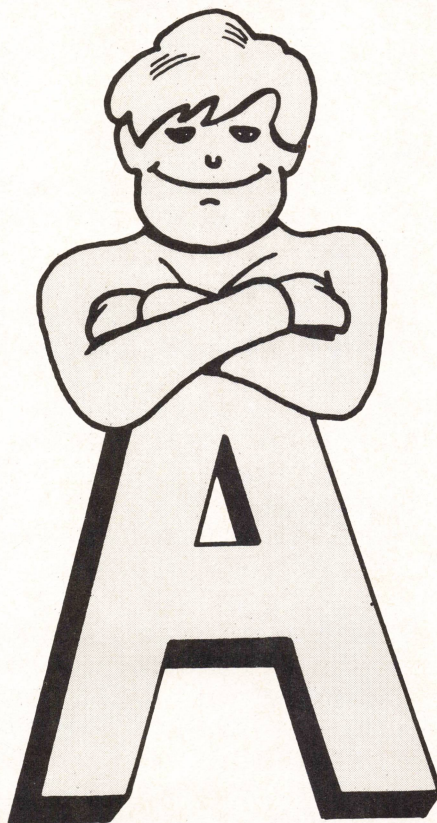
- Load the Accumulator with &2A
- CALL the "print out A" routine at &BB5A
- RETURN from our routine.

Assuming you store your code at &3000, this translates as follows:

address	hex code	mnemonics
3000	3E 2A	LD A, &2A
3002	CD 5A BB	CALL &BB5A
3005	C9	RET

Notice that the address of the routine is in lo byte, hi byte order, and that the routine itself terminates with a RET.

To create the routine via Hexer we must enter the following code (via



option 1) at the default address of &3000:

3E 2A CD 5A BB C9

As you can see, it's simply the hex part of the assembler listing, byte by byte. Examine the code carefully, and, if all is well, run it.

If you now look at the screen you'll see a solitary "unexplained" asterisk – your machine code put it there!

Of course, if you'd loaded the A

register with a different number a different character would have appeared. Suppose for example we'd loaded the accumulator with &41, the assembler listing would look like:

address	hex code	mnemonics
3000	3E 41	LD A, &41
3002	CD 5A BB	CALL &BB5A
3005	C9	RET

The only byte different is that at memory location &3001 – this now holds &41 instead of &2A.

Instead of entering the whole routine again, we can simply alter this byte with option 3 – Alter code. Choose this option, then &3001 as the start address (no need for the preceding "&"). You'll see displayed:

3001 2A ?

confirming that the byte has remained unchanged from its previous value of &2A (Ascii for '*'). As this is the byte we want to change, enter 41 and, after the next prompt, enter S to stop.

If you now examine memory, the code should appear as:

3000 3E 41 CD 5A BB C9

On running the code you'll see an A appear, which isn't surprising as &41 – which you loaded the A register with – is 65 in denary ($4 \times 16 + 1$), the Ascii code for A. Try altering the program so it prints out B, C and so on.

Finally, load A with 7 and CALL &BB5A. Can you explain what happens? If not, try altering the number you load A with to &C and run the program.

If you're still not sure, the answer lies in the table of Ascii codes in chapter 9 of the User Instructions. 7 is Ascii for a bleep and &C is the code for CLS!

Well, this month we've run a few more programs, learned how to use the A register and familiarised ourselves with a powerful tool for working in machine code – Hexer.

Admittedly, the programs aren't all that spectacular, but if you can follow what's going on in them – as I'm sure you can – you're well on your way to learning what machine code is all about.

- Next month more on the A register and its close relatives.

SWAMP!

By
DAVID
MUIR



DEEP in the heart of Oomeetingi Land live the Wotawally tribe, a peaceful race not given to violence and happy to live from one day to the next without a care in the world.

They are ruled by the fearless Chief Amasofti who, over a period of time, has become peeved by a regular invasion of frogs terrorising the village.

You as part of the WAF (Wotawally Against Frogs) campaign have taken it upon yourself to rid the land of this plague for ever. Singlehanded you set off to destroy as many eggs as you can.

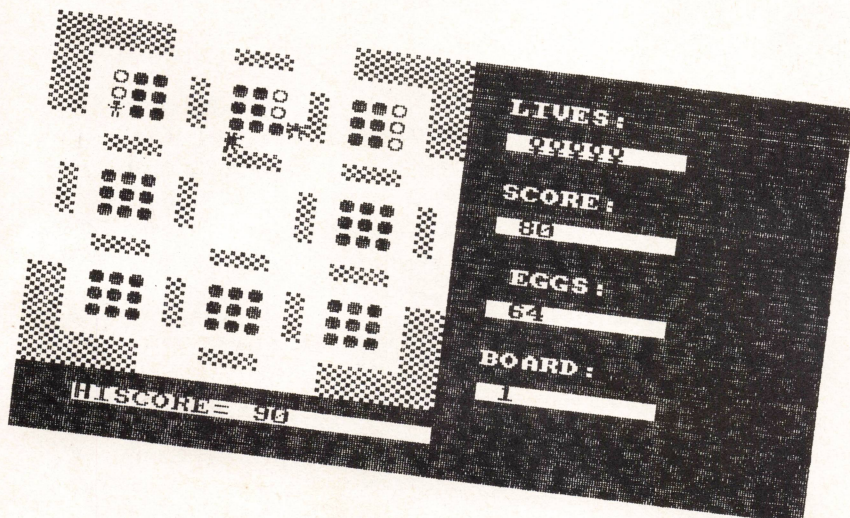
The frogs in this part of the world are a bit special as they are enormous. Quite naturally they were not impressed with your intrusion into the swamp to crush their potential offspring.

To prevent this they have created

```
10 REM
20 REM SWAMP
30 REM
40 REM BY DAVID MUIR
50 REM (C) COMPUTING WITH
60 REM (C) THE AMSTRAD
70 REM
80 DEFINT l,m,n,o,p: DIM l(21,21),m(21,21),nx(1),ny(1),ox(1),oy(1),nc(1),oc(1): hs=0: GOTO 990
90 REM update window displays
100 CLS #2: PRINT #2,sc: RETURN
110 CLS #3: PRINT #3,72-eg: RETURN
120 CLS #1: PRINT #1,LEFT$(t$,li+1): RETURN
130 CLS #4: PRINT #4,bo: RETURN
140 REM check new square
150 IF m(px+dx,py+dy)=254 THEN m(px+dx,py+dy)=255: eg=eg+1: sc=sc+10: GOSUB 100: GOSUB 110: IF eg=72 THEN eg=0: sc=sc+100: bo=bo+1: bf=-1: CLS: li=li+1+1*(li>4): PRINT "WELL DONE!": PRINT: PRINT "YOU MOVE TO ANOTHER": PRINT "PART OF THE SWAMP": GOSUB 190: GOTO 180
160 IF (px=nx(0) AND py=ny(0)) OR (px=nx(1) AND py=ny(1)) THEN li=li-1: bf=-1: CLS: LOCATE 1,1: PRINT "THE FROGS GOT YOU!": eg=0: FOR i=1 TO 4: SOUND 7,600,0,30,5-i,0,0,7-i: NEXT: IF li>0 THEN 180 ELSE 940
170 GOTO 460
180 CLS #1: CLS #2: CLS #3: CLS #4: GOTO
```

```
870
190 FOR i=1 TO 4: SOUND 5,100-i*20,30,i+2: FOR j=1 TO 100: NEXT: NEXT: RETURN
200 REM frogs
210 n=INT(RND*2)
220 lx=SGN(px-nx(n)): ly=SGN(py-ny(n))
230 ds=INT(RND*2+1): ON ds GOTO 300,350
240 nj(n)=nj(n)+1: IF nj(n)=7 THEN nj(n)=0: lx=2*SGN(px-nx(n)): ly=2*SGN(py-ny(n)): GOTO 230
250 GOTO 410
260 nh=-250*(lx>0)-251*(lx<0): IF nh=0 THEN nh=-253*(ly>0)-252*(ly<0)
270 IF nh<>0 THEN nc=nh
280 IF nx(n)+lx=px THEN IF ny(n)+ly=py THEN li=li-1: CLS: LOCATE 1,1: PRINT "THE FROGS GOT YOU!": eg=0: FOR i=1 TO 4: SOUND 7,600,30,5-i,0,0,7-i: NEXT: IF li>0 THEN 180 ELSE 940
290 SOUND 7,600,5,2,0,0,1: LOCATE nx(n),ny(n): PEN 2: PRINT CHR$(m(nx(n),ny(n))): nx(n)=nx(n)+lx: ny(n)=ny(n)+ly: LOCATE nx(n),ny(n): PEN 1: PRINT CHR$(nc): nj(n)=0: GOTO 410
300 IF nx(n)+lx=nx(1-n) AND ny(n)=ny(1-n) THEN 320
310 IF m(nx(n)+lx,ny(n))<>206 AND lx<>0 THEN ly=0: GOTO 260
320 IF nx(n)=nx(1-n) AND ny(n)+ly=ny(1-n) THEN 240
330 IF m(nx(n),ny(n)+ly)<>206 AND ly<
```

```
>0 THEN lx=0: GOTO 260
340 GOTO 240
350 IF nx(n)=nx(1-n) AND ny(n)+ly=ny(1-n) THEN 370
360 IF m(nx(n),ny(n)+ly)<>206 AND ly<>0 THEN lx=0: GOTO 260
370 IF nx(n)+lx=nx(1-n) AND ny(n)=ny(1-n) THEN 240
380 IF m(nx(n)+lx,ny(n))<>206 AND lx<>0 THEN ly=0: GOTO 260
390 GOTO 240
400 REM player move
410 IF INKEY(74)=0 OR INKEY(8)=0 THEN IF px>1 THEN dx=-1: dy=0: GOTO 150
420 IF INKEY(75)=0 OR INKEY(1)=0 THEN IF px<21 THEN dx=1: dy=0: GOTO 150
430 IF INKEY(72)=0 OR INKEY(0)=0 THEN IF py>1 THEN dy=-1: dx=0: GOTO 150
440 IF INKEY(73)=0 OR INKEY(2)=0 THEN IF py<21 THEN dy=1: dx=0: GOTO 150
450 GOTO 210
460 IF m(px+dx,py+dy)=206 THEN 210
470 LOCATE px,py: PEN 2: PRINT CHR$(m(px,py)): SOUND 7,500,6,2
480 px=px+dx: py=py+dy: LOCATE px,py: PEN 3: PRINT CHR$(248): GOTO 210
490 REM graphics
500 REM broken egg
510 SYMBOL 255,0,60,102,66,66,102,60,0
520 REM whole egg
530 SYMBOL 254,0,60,126,126,126,126,6
```

an artificial maze and have posted a couple of bouncers – or perhaps hoppers – to guard it day and night. Your job is to invade the swamp and destroy the eggs by jumping on them.

You move around by using the cursor keys or joysticks, avoiding the two hefty guardians. If you don't it's curtains for one of your five lives.

You score 10 for each egg

destroyed and gain a bonus of 100 plus an extra life if you destroy a whole mazeful of 72. You will also get another swamp to raid.

It won't be easy though, as these giant frogs don't like intruders. So watch it, or else instead of having a frog in your throat you'll be in the throat of a frog!

The program's structure

THE CPC464 has no function in its Basic for determining the character at a particular place on screen. This has been simulated by using the two arrays *l*(21,21) and *m*(21,21).

l() contains the master copy of the screen set up and *m*() contains a second copy which alters as the eggs are destroyed. When a new screen is required *m*() is copied from *l*().

The program contains sufficient REM statements to make it fairly easy to follow.

px & *py* are the player coordinates and *dx* and *dy* are relative movements one square in horizontal or vertical direction.

nx() *ny*() are frog coordinates. *lx* & *ly* are relative movements. *nc* is the character to be used for a frog to jump over the swamp after a certain delay.

hs=hiscore, *sc*=score, *bo*=board number, *li*=lives, *t\$*=string to show lives left and *eg*=eggs broken.

```
0,0
540 REM frog down
550 SYMBOL 253,66,102,36,199,255,60,2
55,153
560 REM frog up
570 SYMBOL 252,153,255,60,255,199,36,
102,66
580 REM frog left
590 SYMBOL 251,216,83,126,248,248,12
,83,216
600 REM frog right
610 SYMBOL 250,27,202,126,31,31,126,2
02,27
620 REM permanent set up
630 MODE 1:BORDER 24:INK 0,18:INK 1,6
,12:INK 2,24:INK 3,2:CLS
640 WINDOW #1,1,40,22,25:PAPER #1,2:CL
S #1
650 WINDOW #1,22,40,1,21:PAPER #1,3:PE
N #1,0:CLS #1
660 sc=0:li=5:eg=0:bo=1:t$=" "+STRING
$(5,235):ni=40
670 PRINT #1:PRINT #1:PRINT #1," LIV
ES:"
680 PRINT #1:PRINT #1:PRINT #1:PRINT
#1:PRINT #1," SCORE:"
690 PRINT #1:PRINT #1:PRINT #1:PRINT
#1:PRINT #1," EGGS:"
700 PRINT #1:PRINT #1:PRINT #1:PRINT
#1:PRINT #1," BOARD:"
710 WINDOW #0,1,21,1,21
720 WINDOW #1,24,32,5,5:PAPER #1,0:PE
```

```
N #1,1
730 WINDOW #2,24,32,10,10:PAPER #2,0
740 WINDOW #3,24,32,15,15:PAPER #3,0
750 WINDOW #4,24,32,20,20:PAPER #4,0
760 WINDOW #5,4,21,23,23:PEN #5,3
770 FOR i=1 TO 21:FOR j=1 TO 21:l(i,j
)=32:NEXT:NEXT
780 FOR h=4 TO 16 STEP 12:FOR i=0 TO
2:FOR j=4 TO 16 STEP 6:FOR k=0 TO 2
790 l(h+i,j+k)=254:l(j+k,h+i)=254:NEX
T:NEXT:NEXT:NEXT
800 FOR h=4 TO 16 STEP 6:FOR i=0 TO 2
:FOR j=2 TO 20 STEP 6
810 l(h+i,j)=206:l(j,h+i)=206:NEXT:NE
XT:NEXT
820 FOR h=4 TO 16 STEP 12:FOR i=0 TO
2:FOR j=1 TO 21 STEP 20
830 l(h+i,j)=206:l(j,h+i)=206:NEXT:NE
XT:NEXT
840 FOR h=1 TO 19 STEP 18:FOR i=0 TO
2:FOR j=1 TO 20 STEP 19:FOR k=0 TO 1
850 l(h+i,j+k)=206:l(j+k,h+i)=206:NEX
T:NEXT:NEXT:NEXT
860 REM set up each screen afresh
870 GOSUB 120:GOSUB 100:GOSUB 110:GOS
UB 130:CLS #5:PRINT #5,"HIScore=";hs;
880 FOR j=1 TO 21:FOR i=1 TO 21:m(i,j
)=l(i,j):PEN 3+l*(m(i,j)=254):LOCATE
i,j:PRINT CHR$(m(i,j));NEXT:NEXT
890 px=1:py=21:nx(0)=10:nx(1)=12:ny(
0)=11:ny(1)=11:nc=253:nj(0)=0:nj(1)=0
:n=0
```

```
900 PEN 1:FOR i=0 TO 1:LOCATE nx(i),n
y(i):PRINT CHR$(nc):NEXT
910 PEN 3:LOCATE px,py:PRINT CHR$(248
)
920 SOUND 7,20,100,2:FOR i=1 TO 500:N
EXT:GOTO 410
930 REM end of game
940 MODE 0:FOR i=1 TO 4:SOUND 5,600+i
*20,30,6-i:FOR j=1 TO 100:NEXT:NEXT:P
RINT " FINAL SCORE":PRINT:PRINT "
";sc:IF sc>hs THEN hs=sc:PRINT:PR
INT " NEW HIScore":FOR i=1 TO 6:S
OUND 5,80-i*10,30,i+2:FOR j=1 TO 100:
NEXT:NEXT
950 PRINT:PRINT " GAME OVER":PRI
NT:PRINT " ANOTHER GO?"
960 q$=INKEY$:IF q$="" THEN 960 ELSE
IF UPPER$(q$)="Y" THEN 630 ELSE IF UP
PER$(q$)<>"N" THEN 960 ELSE CLS:END
970 END
980 REM titles
990 MODE 1:BORDER 12:INK 0,18:INK 1,2
1:CLS
1000 LOCATE 1,11:PRINT SPACE$(9)+STRI
NG$(3,207)+" "+CHR$(207)+" "+CHR$(20
7)+" "+STRING$(4,207)+" "+CHR$(207)+C
HR$(223)+CHR$(222)+CHR$(207)+" "+STRI
NG$(3,207)
1010 PRINT SPACE$(9)+CHR$(207)+" "+
CHR$(207)+" "+CHR$(207)+" "+CHR$(207
)+" "+CHR$(207)+" "+CHR$(207)+CHR$(2
21)+CHR$(220)+CHR$(207)+" "+CHR$(207)
```


Maze Chase

```

+" "+CHR$(207)
1020 PRINT SPACE$(9)+STRING$(3,207)+"
      "+CHR$(207)+" "+CHR$(207)+" "+STRIN
6$(4,207)+" "+CHR$(207)+" "+CHR$(207
)+" "+STRING$(3,207)
1030 PRINT SPACE$(11)+CHR$(207)+" "+C
HR$(207)+CHR$(222)+CHR$(223)+CHR$(207
)+" "+CHR$(207)+" "+CHR$(207)+" "+CH
R$(207)+" "+CHR$(207)+" "+CHR$(207)

```

```

1040 PRINT SPACE$(9)+STRING$(3,207)+"
      "+CHR$(207)+CHR$(220)+CHR$(221)+CHR$
(207)+" "+CHR$(207)+" "+CHR$(207)+"
      "+CHR$(207)+" "+CHR$(207)+" "+CHR$(2
07)

```

```

1050 LOCATE 2,24:PRINT "INSTRUCTIONS?
";

```

```

1060 REM instructions
1070 q$=INKEY$:IF q$="" THEN 1070 ELS
E IF UPPER$(q$)="N" THEN 500 ELSE IF
UPPER$(q$)<>"Y" THEN 1070
1080 CLS:PRINT "SWAMP":PRINT:PRINT "Y
OUR JOB IS TO DESTROY THE EGGS OF 61A

```

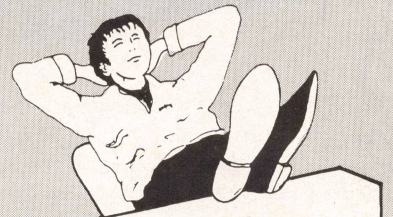


NTFROGS IN THE SWAMP. YOU MOVE USING C
URSORKEYS OR JOYSTICK. YOU DESTROY T
HE SPAWNBY STANDING ON IT.
1090 PRINT:PRINT "YOU HAVE TO AVOID T
HE TWO FROGS. IF THEYCATCH YOU THEN Y
OU WILL LOSE ONE OF YOURFIVE LIVES."

```

1100 PRINT:PRINT "YOU SCORE 10 FOR EA
CH EGG DESTROYED. YOU GAIN A BONUS OF
100 AND 1 EXTRA LIFE FOREACH BOARD OF
72 EGGS YOU DESTROY, AFTERWHICH YOU
ARE GIVEN A NEW SCREEN.
1110 PRINT:PRINT "(PRESS ANY KEY)"
1120 q$=INKEY$:IF q$="" THEN 1120 ELS
E 500
1130 GOTO 500

```



Give your fingers a rest...

All the listings from this month's
issue are available on cassette.
See our special offer on Page 19.

IS PAPER WORK GETTING ON TOP OF YOU ?

ABACUS

BUSINESS SYSTEMS

**CAN BE YOUR
STEPPING STONE**
TO EFFECTIVE FINANCIAL AND
ADMINISTRATIVE CONTROL

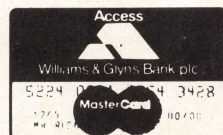
1	PAYROLL	£29.95
2	PURCHASE/SALES LEDGER	£29.95
3	STOCK CONTROL	£17.95
4	NON VAT ACCOUNTS	£17.95
5	CASH PLANNER	£12.95
6	MAILING LIST	£17.95

THE PRICES ABOVE ARE FOR THE CASSETTE VERSION OF
THESE PROGRAMS, DISC VERSIONS USING RANDOM
ACCESS FILES ARE AVAILABLE FROM OCTOBER 1ST 1984.

ALL SOFTWARE PROVIDED BY ABACUS, IS FULLY
SUPPORTED BY THE COMPANY.



21 UNION STREET
RAMSBOTTOM, LANCs
PHONE: 0204 52726



AVAILABLE FROM ALL GOOD COMPUTER SHOPS
AMSTRAD CPC464, BBC
MODEL B, SHARP MZ700
COMMODORE 64
AND DRAGON 32
64

STARFLEET

Have you the potential to become a space pilot? Find out in
ROLAND WADDILOVE's 3D intergalactic flight simulation game

A MESSAGE has been received from Star Fleet Command that they are in need of keen new pilots to protect the Earth's inner space territory. There has been a mass exodus of the present corps to join Swissair – apparently the pay is better.

Requirements dictate fitness, good eyesight, quick reactions and a willingness to work long hours. (*It's a bit like working here really – Editor.*)

In a fit of excess patriotism and greed – the perks are good – you volunteer and as a result have been requested to undertake a test mission. This will be accomplished using the Fleet's own intergalactic flight simulator.

During the test you must take

on the Thargoid fleet in three different locations. As you warp through space you will see each enemy ship approaching from the distance.

You control the movement of the simulator using either joysticks or the following keys to simulate joystick movement:

- A to dive
- Z to climb
- < to bank left
- > to bank right

In order to destroy the enemy craft you must position it within the cross sight and fire your lasers using either the joystick fire button or the space bar.

You will not be able to harm the enemy until he comes into range, and he cannot return your fire until you are within his rather more limited range.

His fire automatically homes in on your ship and reduces your

power relative to the number of times you are hit.

There is no room for error and you must be spot on target to destroy him and score points. Should you survive the first wave you will encounter a second and if successful a third.

Your acceptance rating will be assessed on your performance. The final decision as to your future is displayed once you have lost all power and the simulation is terminated.

If you fail to qualify in the early stages don't be dismayed. Even some of the top Star Fleet pilots couldn't hit a barn door from 10 paces when they first applied, so there's chance for all of us. Good hunting!




```

10 REM *****
20 REM *   Star Fleet   *
30 REM * By R.A.Waddilove *
40 REM * (C) Computing *
50 REM * with the Amstrad *
60 REM *****
70 MODE 1
80 GOSUB 1900 :REM instructions
90 MODE 0
100 GOSUB 270 :REM initialise
110 WHILE 1
120 GOSUB 380 :REM start
130 WHILE power%>530
140 GOSUB 710:IF INKEY(e%)>-1 THEN GO
SUB 780 ELSE GOSUB 1620
150 WEND
160 GOSUB 1670 :REM game over
170 WEND
180 MODE 1
190 CALL &BB4E:CALL &BC02:REM reset v
du
200 END
210 REM
220 REM ***** move ground *****
230 g%=(g%+1)MOD 3:CALL &BD19:IF g%=0
THEN INK 0,0:INK 2,colour%:RETURN
240 IF g%=1 THEN INK 0,colour%:INK 1,
0:RETURN
250 INK 1,colour%:INK 2,0:RETURN
260 REM
270 REM ***** initialise *****
280 ENT 1,100,10,1:ENT 2,1,0,2,100,10
,1:ENT -3,5,10,1,5,-10,1
290 SYMBOL AFTER 250
300 SYMBOL 255,24,36,189,255,255,189,
24,36
310 SYMBOL 254,0,16,170,254,254,186,1
6,0
320 SYMBOL 253,0,0,90,126,90,0,0,0
330 SYMBOL 252,0,0,24,60,24,0,0,0
340 SYMBOL 251,0,0,0,24,24,0,0,0
350 SYMBOL 250,0,0,0,16,0,0,0,0
360 RETURN
370 REM
380 REM ***** start *****
390 DATA 0,0,0,0,2,24,15,0,26,26,26,2
6,2,24,15,15
400 RESTORE 390
410 FOR i%=0 TO 15
420 READ j%:INK i%,j%
430 NEXT
440 screen=1:colour%=1
450 power%=630:score%=0
460 PRINT #1,CHR$(23);CHR$(0)
470 BORDER 1:PAPER 4:CLS
480 PEN 5:LOCATE 1,25:PRINT "Score:0

```

```

Power";
490 MOVE 530,8:DRAW 630,8,5
500 MOVE 530,6:DRAW 630,6
510 ORIGIN 320,206,8,630,16,394:CLS 3
520 TAG
530 PEN #2,5:PAPER #2,4
540 EVERY 12 GOSUB 230
550 GOSUB 950 :REM draw screen
560 GOSUB 590 :REM new ship
570 RETURN
580 REM
590 REM ***** new ship *****
600 PRINT #1,CHR$(23);CHR$(1)
610 PLOT 0,0,8
620 RANDOMIZE TIME
630 x%=200+RND*200:y%=150+RND*100
640 s%=250:h%=0:v%=0:count%=0
650 FOR i%=1 TO 5000:NEXT
660 MOVE x%,y%:PRINT CHR$(s%);
670 SOUND 132,2000,2000,1,0,3
680 RETURN
690 REM
700 REM ***** move ship *****
710 PLOT 0,0,8:IF s%<255 THEN count%=
count%+1:IF count%=50 THEN count%=0:s
%=s%+1:MOVE x%,y%:PRINT CHR$(s%-1);M
OVE x%,y%:PRINT CHR$(s%);:SOUND 132,2
000,32000,s%-250,0,3
720 IF RND>0.5 THEN h%=INT(RND*17)-8:
v%=INT(RND*9)-4
730 xx%=x%+h%-4*((INKEY(a%))>-1)-(INKE
Y(b%))>-1):yy%=y%+v%-2*((INKEY(c%))>-1
)-(INKEY(d%))>-1):IF (xx%<0 OR xx%>64
0) OR (yy%<0 OR yy%>400) THEN RETURN
740 CALL &BD19:MOVE x%,y%:PRINT CHR$(
s%);:MOVE xx%,yy%:PRINT CHR$(s%);:x%=
xx%:y%=yy%
750 RETURN
760 REM
770 REM ***** fire *****
780 MOVE 225,50:DRAW 320,208:DRAW 422
,50:SOUND 129,50,40,15,0,1:SOUND 130,
50,40,15,0,2:SOUND 1,1000,32000,3,0,3
:PRINT #1,CHR$(30);CHR$(23);CHR$(0):P
LOT power%,8,4:PLOT power%,6:PRINT #1
,CHR$(23);CHR$(1):power%=power%-1
790 MOVE 422,50:DRAW 320,208,8:DRAW 2
25,50:IF TEST(320,206)=3 THEN RETURN
800 score%=score%+10*(256-s%):LOCATE
#2,7,25:IF score%>999 THEN PRINT #2,U
SING "###";score%; ELSE IF score%>99
THEN PRINT #2,USING "##";score%; EL
SE PRINT #2,USING "#";score%;
810 SOUND 132,0,400,7,0,1,1
820 xx%=x%-16:yy%=y%:ex%=x%+16:ey%=y%
830 MOVE x%,y%:PRINT CHR$(s%);

```

```

840 FOR i%=1 TO 25
850 MOVE x%,y%:PRINT CHR$(144);:MOVE
xx%,yy%:PRINT CHR$(144);:MOVE ex%,ey%
:PRINT CHR$(144);
860 FOR k%=1 TO 100:NEXT
870 MOVE x%,y%:PRINT CHR$(144);:MOVE
xx%,yy%:PRINT CHR$(144);:MOVE ex%,ey%
:PRINT CHR$(144);
880 y%=y%+i%:xx%=xx%-i%:yy%=yy%-i%\2:
ex%=ex%+i%:ey%=ey%-i%\2
890 NEXT
900 IF score%>249 AND screen=1 THEN s
creen=2:colour%=3:GOSUB 950
910 IF score%>499 AND screen=2 THEN s
creen=3:colour%=9:GOSUB 950
920 GOSUB 590 :REM new ship
930 RETURN
940 REM
950 REM ***** draw screen *****
960 SOUND 129,0,1,0:SOUND 130,0,1,0:S
OUND 132,0,1,0
970 LOCATE #1,7,7:PEN #1,8:PRINT #1,C
HR$(22);CHR$(1);"Screen";screen;CHR$(
22);CHR$(0)
980 GOSUB 1490:FOR i%=0 TO 5000:NEXT
990 PRINT #1,CHR$(23);CHR$(0)
1000 ORIGIN 320,206,8,630,16,394:CLS
3
1010 FOR i%=1 TO 25:PLOT 200+RND-100,
200+RND-100,3+RND:NEXT:PLOT 0,0,3
1020 MOVE 0,20:DRAW 0,32,4:MOVE 20,0:
DRAW 32,0:MOVE 0,-20:DRAW 0,-32:MOVE
-20,0:DRAW -32,0
1030 ON screen GOSUB 1100,1370,1210
1040 ORIGIN 0,0,700,800,700,700:CLS 0
:ORIGIN 0,0,0,640,0,400
1050 MOVE 196,16:DRAW 216,50,4:DRAW 2
20,50:DRAW 200,16:DRAW 204,16:DRAW 22
0,50
1060 MOVE 444,16:DRAW 426,50:DRAW 422
,50:DRAW 440,16:DRAW 436,16:DRAW 426,
50
1070 SOUND 1,1000,32000,3,0,3
1080 RETURN
1090 REM
1100 REM ***** screen 1 *****
1110 x%=90:y%=70
1120 FOR i%=1 TO 20
1130 FOR j%=0 TO i%\4
1140 MOVE x%,y%:DRAW -x%,y%,i% MOD 3:
DRAW -x%,-y%:DRAW x%,-y%:DRAW x%,y%
1150 x%=x%+4:y%=y%+2
1160 NEXT
1170 NEXT
1180 MOVE 90,70:DRAW 420,235,3:MOVE -
90,70:DRAW -420,235:MOVE -90,-70:DRAW

```



```

-420,-235:MOVE 90,-70:DRAW 420,-235
1190 RETURN
1200 REM
1210 REM ***** screen 3 *****
1220 DEG:COLX=0:XL=100
1230 FOR IX=1 TO 35
1240 COLX=(COLX+1) MOD 3
1250 FOR KX=0 TO IX\4
1260 XL=XL+2:PLOT XL,0,COLX
1270 FOR JX=72 TO 360 STEP 72
1280 DRAW XL*COS(JX),XL*SIN(JX)
1290 NEXT
1300 NEXT
1310 NEXT
1320 FOR JX=72 TO 360 STEP 72
1330 MOVE 100*COS(JX),100*SIN(JX):DRA
W 400*COS(JX),400*SIN(JX),3
1340 NEXT
1350 RETURN
1360 REM
1370 REM ***** screen 2 *****
1380 COLX=0:XL=100
1390 FOR IX=1 TO 33
1400 COLX=(COLX+1) MOD 3
1410 FOR JX=0 TO IX\5
1420 MOVE XL,-XL\2:DRAW 0,XL,COLX:DRA
W -XL,-XL\2:DRAW XL,-XL\2
1430 XL=XL+4
1440 NEXT
1450 NEXT
1460 MOVE 0,100:DRAW 0,200,3:MOVE 100
,-50:DRAW 320,-160:MOVE -320,-160:DRA
W -100,-50
1470 RETURN
1480 REM
1490 REM ***** tune *****
1500 SOUND 1,478,100,6
1510 SOUND 2,379,50,0:SOUND 2,379,50,
6
1520 SOUND 4,319,75,0:SOUND 4,319,25,
6
1530 SOUND 1,506,100,7
1540 SOUND 2,426,25,0:SOUND 2,426,75,
6
1550 SOUND 4,319,50,0:SOUND 4,319,50,
6
1560 SOUND 1,568,300,6
1570 SOUND 2,379,300,6
1580 SOUND 4,284,100,0:SOUND 4,284,25
,6:SOUND 4,239,50,6:SOUND 4,190,125,6
1590 RETURN
1600 REM
1610 REM ***** ship fires back *****
1620 IF SX<255 THEN FOR IX=1 TO 60:NE
XT:RETURN
1630 IF RND<0.9 THEN RETURN

```

```

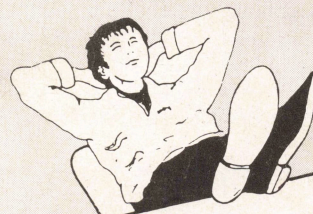
1640 INK 4,17:INK 3,13:SOUND 130,5,10
,15,0,2:MOVE XL,YL:DRAW 320,16,6:DRAW
XL+28,YL:PRINT #1,CHR$(30);CHR$(23);
CHR$(0):PLOT powerXL,0,4:PLOT powerXL,6
:PRINT #1,CHR$(23);CHR$(1):powerXL=pow
erXL-1
1650 MOVE XL+28,YL:DRAW 320,16,6:DRAW
XL,YL:INK 4,2:INK 3,0:RETURN
1660 REM
1670 REM ***** game over *****
1680 SOUND 129,0,1,0:SOUND 130,0,1,0:
SOUND 132,0,1,0
1690 LOCATE #1,4,7:PEN #1,8:PRINT #1,
CHR$(22);CHR$(1);"SIMULATION OVER";CH
R$(22);CHR$(0)
1700 GOSUB 1490:FOR IX=0 TO 10000:NEX
T
1710 IX=REMAIN(0)
1720 PAPER 0:PEN 1:INK 0,0:MODE 1:BOR
DER 0
1730 INK 1,19:INK 2,2:INK 3,7
1740 PRINT SPC(11)"SIMULATOR RATING"
1750 PEN 2:LOCATE 9,6:PRINT "Your fin
al score was";scoreXL
1760 LOCATE 4,9:PRINT "This is consid
ered to be ";
1770 scoreXL=scoreXL\100
1780 IF scoreXL=0 THEN PRINT "abysmal.
":a$="Your application has been rejec
ted."
1790 IF scoreXL=1 THEN PRINT "very poo
r.":a$="You need more flying lessons.
"
1800 IF scoreXL=2 THEN PRINT "average.
":a$="You have just scraped through."
1810 IF scoreXL=3 THEN PRINT "quite go
od.":a$="Your application has been ac
cepted."
1820 IF scoreXL=4 THEN PRINT "very goo
d.":a$=SPACE$(9)+"Welcome to Star Fle
et."
1830 PRINT:PRINT SPC((40-LEN(a$))/2)a
$
1840 WHILE INKEY$<>"":WEND
1850 LOCATE 1,20:PEN 3:PRINT "Press S
PACE or FIRE for another battle."
1860 WHILE INKEY$<>" " AND INKEY(76)=
-1:WEND
1870 MODE 0
1880 RETURN
1890 REM
1900 REM ***** instructions *****
1910 CALL &BB4E:CALL &BC02:REM reset
vdu
1920 INK 0,0: BORDER 0:INK 1,21

```

```

1930 DATA HCHIJKJAHKAHKBKLB
1940 DATA LAFKLKJAHKAHKBKA
1950 DATA DABCCCCABCBDCBCA
1960 RESTORE 1930
1970 FOR IX=1 TO 3
1980 READ a$:PRINT SPC(12)
1990 FOR JX=1 TO LEN(a$)
2000 PRINT CHR$(ASC(MID$(a$,JX,1))+63
);
2010 NEXT
2020 PRINT
2030 NEXT
2040 GOSUB 1490
2050 PEN 2:PRINT:PRINT:PRINT " Star F
leet Application No. #5638-D6":PRINT
" Notes for applicants..."
2060 PEN 1:PRINT:PRINT:PRINT " Only t
he best pilots are accepted by Sta
r Fleet. All applicants must prove
themselves worthy by fighting Thargoi
d"
2070 PRINT " ships in the flight simu
lator. If they do not score high eno
ugh then they are rejected."
2080 PEN 2:PRINT:PRINT:PRINT " Use jo
ystick or..."
2090 PEN 1:PRINT:PRINT " A Z < >
and SPACE to fire."
2100 PEN 3:PRINT:PRINT:PRINT " Prepar
e yourself for battle, then":PRINT "
press fire when you are ready."
2110 GOSUB 1490:WHILE INKEY$<>"":WEND
2120 ok=1
2130 WHILE ok
2140 IF INKEY(47)>-1 THEN ok=0:aZ=39:
bZ=31:cZ=69:dZ=71:eZ=47
2150 IF INKEY(76)>-1 THEN ok=0:aZ=74:
bZ=75:cZ=72:dZ=73:eZ=76
2160 WEND
2170 RETURN

```



Give your fingers a rest...

All the listings from this month's issue are available on cassette. See our special offer on Page 19.

I F you've been following the series so far, by now you should be familiar with our old favourite:

`SOUND 1,200,100,5`

Hopefully you'll be able to see that this tells the Amstrad to make a sound on channel A that lasts for one second. The pitch of the note will be 200 and its volume will be 5.

As you'll recall, the `SOUND` command has the structure:

`SOUND channel,pitch,duration,volume`

and by altering these parameters we alter the resulting noise.

Things are never quite that simple and last month we saw that the volume of the note played could be changed by something called a volume envelope.

We can have 15 of these volume envelopes, defined by the `ENV` command and called up by attaching another parameter to the end of our basic `SOUND` statement.

So, by combining:

`ENV 1,5,2,20`

and:

`SOUND 1,200,100,5,1`

we get a note that lasts for one second, its volume getting louder as it plays.

The structure of the `ENV` command is:

`ENV N,P,Q,R`

where N just labels the envelope, P

Pitch in ... and give your tunes some tone

gives the number of steps, Q the volume change per step and R specifies how long each step will last.

Again however, things are never quite that simple and we saw that the `ENV` command could take up to 16 parameters in the form:

`ENV N,P1,Q1,R1,P2,Q2,R2,
P3,Q3,R3,P4,Q4,R4,
P5,Q5,R5`

This surfeit of parameters allows the volume envelope to have up to five stages. As if all this wasn't enough, the volume envelope isn't the only envelope that can affect our basic `SOUND` command.

There's another envelope called

the pitch – or tone – envelope which affects the pitch of the note – how high or low it sounds. Before we go into how it works, let's hear it in action.

First, define a pitch envelope with:

`ENT 1,5,10,20`

Next type in:

`SOUND 1,200,100,5,0,1`

and press Enter.

If you've typed it all in correctly you should hear a noise that lasts for one second, getting lower and lower in pitch.

What's happened is that the final 1 in the `SOUND` command has called the pitch envelope labelled 1. This

	Channel	Pitch	Duration	Volume		Volume Envelope	Pitch Envelope
				without envelope	with envelope		
Range	1=A 2=B 4=C	0 to 4095	1 to 32767	0 to 7	0 to 15	0 to 15	0 to 15
Default	none	none	20	4	12	0	0

Table I: Parameter ranges for `SOUND` command

Parameter	Number S	Number of steps in section T	Pitch change per step V	Time length of each step W
Range	0 to 15	0 to 239	-128 to 127	0 to 255

Table II: Parameter ranges for `ENT` command

previously-defined envelope then varies the pitch of the note produced by the SOUND statement in line with the pitch envelope's parameters.

You'll notice that we now have six parameters following the SOUND command. Table I shows the new parameter ranges for the SOUND command.

As you can see, the pitch envelope looks very similar to the volume envelope we dealt with previously. It takes the form:

ENT S,T,V,W

and as you might guess, S is just a number that labels the pitch envelope. You can define up to 15 of these pitch envelopes so S ranges from 1 to 15. A value of 0 leaves the note unchanged.

The T, V and W parameters again mimic those in the volume envelope but in this case they affect how the highness or lowness of the note varies, not its loudness.

The T parameter decides on the number of steps there are going to be in the pitch envelope. It can have values between 0 and 239.

The V parameter is the one that decides how much the pitch is going to vary at each step. The pitch can go either up or down, taking values between -128 and 127.

Finally the W parameter decides how long each step is to last. Measured in hundredths of a second, it can take values between 0 and 255. Table II sums up the parameters of the ENT command and the values they can take.

Now that we know what these parameters do, let's see how they worked on our old favourite sound. Figure I shows diagrammatically the pitch of the note produced by:

SOUND 1,200,100,5

As you can see, the pitch stays steady at 200 for the second that the note lasts.

Now let's define a pitch envelope with:

ENT 1,5,10,20

and call it up with:

SOUND 1,200,100,5,0,1

As you'll hear, the sound descends in pitch in five steps during the second that it plays. Figure II shows the five steps of the pitch envelope

Part III of NIGEL PETERS' series on coaxing melodious sounds from the CPC464

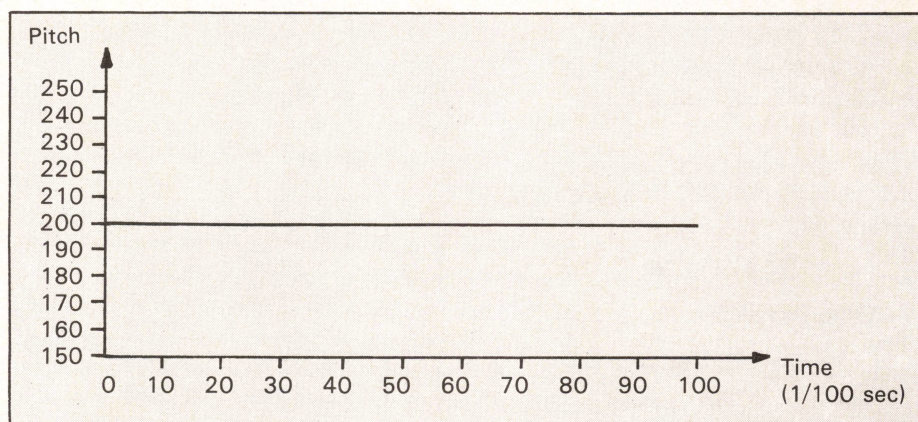


Figure I: SOUND 1,200,100,5

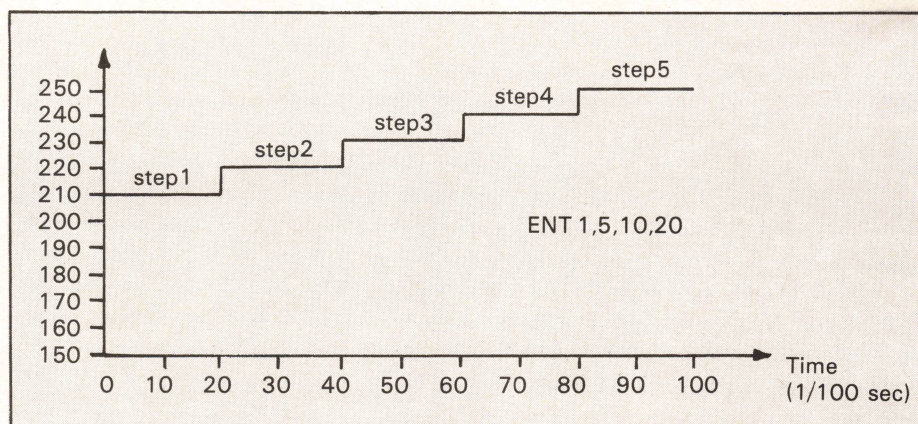


Figure II: SOUND 1,200,100,5,0,1

graphically.

Let's take a look at this pitch envelope in detail. The T parameter is 5, ensuring that there will be five steps, while the W parameter of 20 ensures that each step will last for a fifth of a second.

The V parameter of 10 means that at each step 10 is added to the pitch of the note that is playing. In the case of:

SOUND 1,200,100,5,0,1

this means that there will be five

notes played with pitches of 210, 220, 230, 240 and 250. The envelope takes the pitch parameter of 200 from the SOUND command and successively adds 10 to it. As the value of the pitch parameter increases, so the note gets lower.

Notice that the pitch is incremented straight away – the sound starts at pitch 210, not 200 as you might expect. The pitch envelope takes effect immediately. Also notice that a single SOUND command has produced five notes courtesy of a previously-defined pitch envelope.

Before this we would have had to use five SOUND commands to get the same effect, as in Program I.

```
10 REM Program I
20 SOUND 1,210,20,5
30 SOUND 1,220,20,5
40 SOUND 1,230,20,5
50 SOUND 1,240,20,5
60 SOUND 1,250,20,5
```

Now however, we can get the same result by defining a pitch envelope with:

```
ENT 1,5,10,20
```

and calling it using:

```
SOUND 1,200,100,5,0,1
```

which is a lot simpler. And the same envelope can be used to vary the pitch of other notes in the same way. Try:

```
SOUND 1,100,100,5,0,1
```

which calls the same pitch envelope but starts at a higher pitch (110).

To sum up so far, we can define a pitch envelope using ENT. When this is called, it alters the pitch of the sound produced by a SOUND command.

In case you're wondering, you can have both volume and pitch envelopes operating at the same time. Try:

```
SOUND 1,200,100,5,1,1
```

and – unless you've cleared the envelopes out of your micro and have to re-enter them – you'll hear five descending notes getting louder as they get lower. The volume and pitch envelopes are working in unison.

As I said before, you can have up to

15 pitch envelopes so let's define another one with:

```
ENT 2,5,-10,20
```

Can you guess what its effect will be before you try it out on a SOUND command?

The T parameter is 5, so there will be five steps. Since the W parameter is 20, this means that each step will last for 20 hundredths of a second. The V parameter is -10 so the value of the pitch parameter will decrease by 10 for each step of the pitch envelope.

As the pitch parameter decreases in value, so the note paradoxically gets higher in pitch. So we'll get a note lasting one second, increasing in pitch by five stages. Call the envelope with:

```
SOUND 1,200,100,5,0,2
```

and hear for yourself.

Again, one simple pitch envelope has produced five notes of different pitch. If we didn't use an envelope we would have to resort to something like Program II to achieve our aims.

```
10 REM Program II
20 SOUND 1,190,20,5
30 SOUND 1,180,20,5
40 SOUND 1,170,20,5
50 SOUND 1,160,20,5
60 SOUND 1,150,20,5
```

As you can see:

```
SOUND 1,200,100,5,0,2
```

is much easier.

You'll probably have noticed that the pitch envelope expects the sound to last a certain time. So far our examples have always had the SOUND command last that amount of time. Suppose we defined a pitch envelope with:

```
ENT 3,5,20,40
```

As you can see from the T and W parameters, the envelope expects that there will be five steps and that each step will last 40 hundredths of a second. That means the whole pitch envelope will last two seconds.

But suppose the SOUND command that invokes the pitch envelope only has a duration parameter of a second? In other words, the duration of the SOUND command is less than

that assumed in the pitch envelope. What happens?

As with most things in computing, the answer is to try it and see. Entering:

```
SOUND 1,200,100,5,0,3
```

will give you the answer. The noise still lasts only one second. The pitch envelope only gets through two and a half steps before it's cut off in its prime.

```
SOUND 1,200,200,5,0,3
```

which lasts two seconds, will let you hear all of the envelope's effects.

But what of the other case, where the pitch envelope lasts for a shorter time than the SOUND command? Enter:

```
ENT 4,5,-10,10
```

which defines a pitch envelope that expects to last half a second. Now call this newly-created envelope with:

```
SOUND 1,200,100,5,0,4
```

which should last one second.

As you can hear, the pitch envelope lasts for its full half second, the note rising in pitch. Then for the remaining half second the note remains at the final pitch.

The envelope has its way and then the SOUND command uses up the remaining time playing at the final pitch.

One other problem that might crop up is where the V parameter of a pitch envelope tries to take the pitch out of range.

As we know, the value of the pitch parameter can only range from 0 to 4095. So what happens if the increase or decrease of pitch in one of the envelope's steps tries to take it out of this range?

When we came across a similar problem in the volume envelope we saw that the Amstrad just wrapped round to values that were in range. This is also the case with the pitch envelope. Try:

```
ENT 5,5,-100,100
```

```
SOUND 1,300,500,5,0,5
```

and:

```
ENT 6,5,100,100
```

```
SOUND 1,3800,500,5,0,6
```

and you'll hear what I mean. The silent part occurs when the pitch

parameter is equal to zero.

And that's about all for this month except to inform you that, as ever, the pitch envelope isn't as simple as I've made it seem. Like the volume envelope it can have up to five sections instead of just the one we've been using so far.

This means that instead of:

```
ENT S,T,V,W
```

the actual definition of a pitch envelope is:

```
ENT S,T1,V1,W1,T2,V2,W2,
T3,V3,W3,T4,V4,W4,
T5,V5,W5
```

Once again we've got a huge beast with 16 parameters. And once again let me tell you that it's not as bad as it looks.

Although we've got five sections each behaves exactly the way as the first one we've been looking at. The difference is that instead of T, V and W the first section has parameters T1, V1 and W1, the second T2, V2 and W2 and so on. Figure III shows how the parameters relate to the sections.

Although you can have five sections in a pitch envelope — as should be obvious from the above — you don't have to have all five in use. For illustration let's take a pitch envelope with three sections, such as the one defined with:

```
ENT 1,5,10,20,5,-5,20,5,5,20
```

This pitch envelope has the label 1 and is in three sections lasting a total of three seconds. Taking each section in turn you should be able to see what happens. When you think you've figured it out call the envelope with:

```
SOUND 1,200,300,5,0,1
```

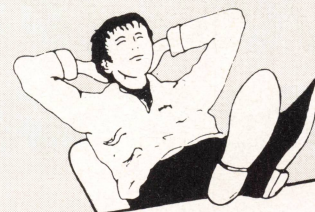
and see if you were right.

Don't be worried by all the

parameters of the pitch envelope. So long as you don't let it know you're afraid of it you'll be all right.

And to give you practice I leave you with Program III to help create your own pitch envelopes and hear what they're like.

```
10 REM PROGRAM III
20 REM TONE ENVELOPE
30 DIM T(5),V(5),W(5)
40 WHILE -1
50 MODE 1
60 INPUT "How many sections in tone e
nvelope?", sections
70 IF sections<1 OR sections >5 THEN
CLS:GOTO 60
80 CLS
90 FOR loop=1 TO sections
100 LOCATE 3,5:PRINT "Section" loop
110 LOCATE 3,8:PRINT "Number of steps
?"
120 LOCATE 30,8:INPUT T(loop)
130 IF T(loop)<0 OR T(loop)>239 THEN
LOCATE 30,8:PRINT SPACE$(8):GOTO 120
140 LOCATE 3,13:PRINT "Size of each s
tep?"
150 LOCATE 30,13:INPUT V(loop)
160 IF V(loop)<-128 OR V(loop)>127 TH
EN LOCATE 30,13:PRINT SPACE$(8):GOTO
150
170 LOCATE 3,18:PRINT "Duration of st
ep?"
180 LOCATE 30,18:INPUT W(loop)
190 IF W(loop)<0 OR W(loop)>255 THEN
LOCATE 30,18:PRINT SPACE$(8):GOTO 180
200 LOCATE 14,23:PRINT "PRESS SPACE"
210 WHILE INKEY(47)=-1:WEND:CLS
220 WHILE INKEY<>"":WEND
230 NEXT loop
240 ENT 1,T(1),V(1),W(1),T(2),V(2),W(
2),T(3),V(3),W(3),T(4),V(4),W(4),T(5)
,V(5),W(5)
250 duration=T(1)*W(1)+T(2)*W(2)+T(3)
*W(3)+T(4)*W(4)+T(5)*W(5)
260 SOUND 1,200,duration,5,0,1
270 CLS
280 duration$=RIGHT$(STR$(duration),L
EN(STR$(duration))-1)
290 PRINT "SOUND 1,200,";duration$;
,5,0,1"
300 FOR loop=1 TO sections
310 loop$=RIGHT$(STR$(loop),1)
320 PRINT "T(";loop$;)" ";T(loop)
330 PRINT "V(";loop$;)" ";V(loop)
340 PRINT "W(";loop$;)" ";W(loop)
350 NEXT
360 LOCATE 14,23:PRINT "PRESS SPACE"
370 WHILE INKEY(47)=-1:WEND:CLS
380 WEND
```



Give your fingers a rest . . .

All the listings from this month's issue are available on cassette. See our special offer on Page 19.

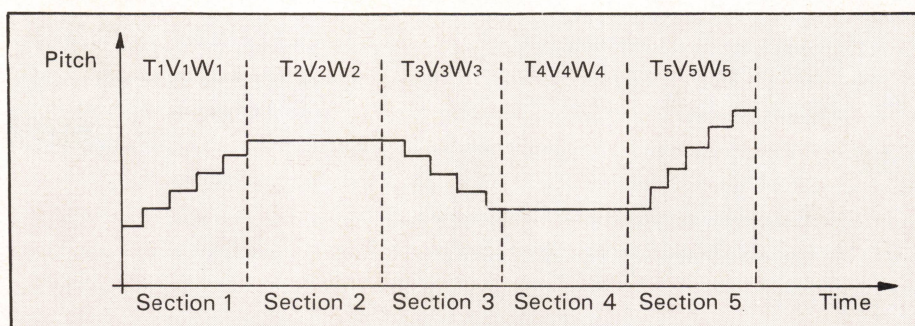


Figure III: Parameters for all five sections of a pitch envelope

And that's all for this month, though it's not the end of our treatment of envelopes. After all, there are some very important questions needing an answer. Such as why have envelopes in the first place?

QUICK TO LEARN

THAT'S...

MINI MOFFICE

**JUST LOOK WHAT THIS
PACKAGE CAN DO!**

WORD PROCESSOR – Ideal for writing letters or reports! *Features:* Constant time display ● Constant word count (even shows words per minute) ● Normal or double-height text on screen or printout.

SPREADSHEET – Use your micro to manage your money! *Features:* Number display in rows and columns ● Continuous updating ● Update instantly reflected throughout spreadsheet ● Save results for future amendments.

GRAPHICS – Turn those numbers into an exciting visual display! *Features:* 3D bar chart ● Pie chart ● Graph.

DATABASE – Use it like an office filing cabinet! *Features:* Retrieve files at a keystroke ● Sort ● Replace ● Save ● Print ● Search.

SPREADSHEET

	A	B	C	D
1	MONEY	JANUARY	FEBRUARY	MARCH
2	MORTGAGE	85.72	85.72	85.72
3	FOOD	46.24	41.43	36.45
4	FUEL	46.25	47.28	36.28
5	LEISURE	20.00	20.00	20.00
6	OTHER	99.85	17.12	56.23
8	TOT SPENT	298.06	211.55	234.68
11	EARNINGS	321.21	321.21	321.21
12	B. FWD.	27.25	0.00	27.41
14	TO SPEND	348.46	321.21	348.62
15	SPENT	298.06	211.55	234.68
17	REMAINING	0.00	109.66	113.95
19	SAVE	0.00	82.35	85.46
20	C.FWD.	0.00	27.41	28.49

DATABASE

RECORD No. 1
SURNAME: JONES
FIRST NAME: SIMON
ADDRESS1: 6 BROAD LANE
ADDRESS2: LIVERPOOL
TELEPHONE: 051-633 8000
AGE: 42

RECORD No. 1
SURNAME: ANDREWS
FIRST NAME: JAMES
ADDRESS1: 12 ELF ROAD
ADDRESS2: HEREFORD
TELEPHONE: 321-623451
AGE: 13

RECORD No. 2
SURNAME: ANDREWS
FIRST NAME: PETER
ADDRESS1: 12 ELF ROAD
ADDRESS2: HEREFORD
TELEPHONE: 321-623451
AGE: 19

RECORD No. 2
SURNAME: ANDREWS
FIRST NAME: PETER
ADDRESS1: 12 ELF ROAD
ADDRESS2: HEREFORD
TELEPHONE: 321-623451
AGE: 19

RECORD No. 3
SURNAME: SMITH
FIRST NAME: JANE
ADDRESS1: 42 HIGH STREET
ADDRESS2: SALFORD
TELEPHONE: 823-61421
AGE: 27

RECORD No. 3
SURNAME: BRINN
FIRST NAME: KIETH
ADDRESS1: 15 MILL ROAD
ADDRESS2: WARRINGTON
TELEPHONE: 853-80923
AGE: 30

RECORD No. 4
SURNAME: YATES
FIRST NAME: IAN
ADDRESS1: 177 FORD ROAD
ADDRESS2: GULLHAM
TELEPHONE: 452-986 76543
AGE: 35

RECORD No. 4
SURNAME: BROWN
FIRST NAME: IAN
ADDRESS1: 17 LEAWARD
ADDRESS2: NORWICH
TELEPHONE: 831-34381
AGE: 21

RECORD No. 5
SURNAME: ANDREWS
FIRST NAME: JAMES
ADDRESS1: 12 ELF ROAD
ADDRESS2: HEREFORD
TELEPHONE: 321-623451
AGE: 13

RECORD No. 5
SURNAME: BROWN
FIRST NAME: JIM
ADDRESS1: 8 ELM RD
ADDRESS2: NANTWICH
TELEPHONE: 681-458
AGE: 11

...and it's all at
price of just

...N, EASY TO USE

	F	G	H	I	J	K	L	M	N	O
APRIL	MAY	JUNE	JULY	AUGUST	SEPTEMBER	OCTOBER	NOVEMBER	DECEMBER		TOTAL
85.72	91.37	91.37	91.37	91.37	91.37	85.34	85.34	85.34		1055.75
32.71	41.23	38.29	32.71	27.98	35.99	40.89	39.89	46.45		460.26
32.61	25.41	20.04	22.34	16.85	24.96	29.77	35.55	48.23		385.57
25.00	25.00	25.00	25.00	25.00	30.00	30.00	30.00	30.00		305.00
100.87	49.29	16.45	29.96	19.49	36.89	107.90	38.02	79.49		651.56
	232.30	191.15	201.38	180.69	219.21	293.90	228.80	289.51		2858.14
										4111.32
										25.40
										4176.72
										2858.14
										1278.58
										958.94
										719.65

*Specially written
for your*
AMSTRAD CPC 464

GRAPHICS

WORD PROCESSOR

Page 1

This is a demonstration of the
MINI OFFICE word processor
showing the various printout
options available.

Page 1

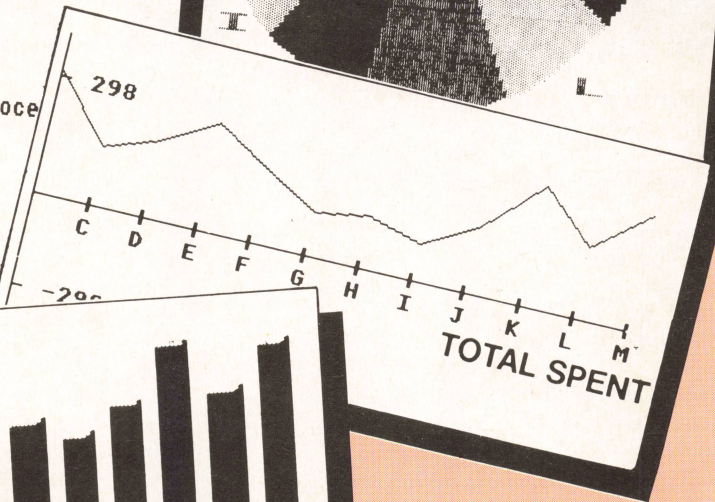
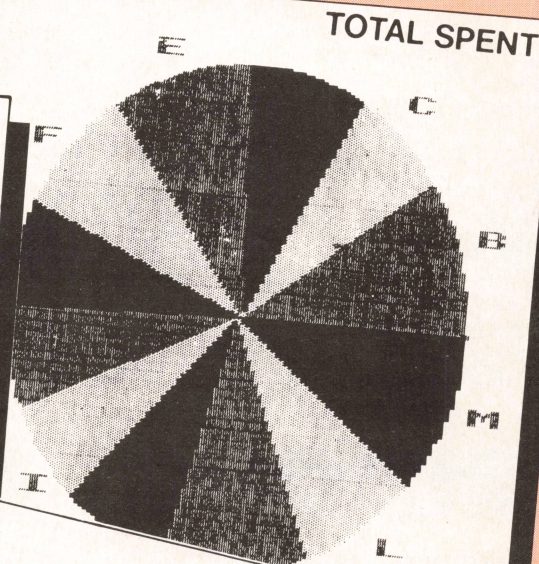
This is a demonstration of the MINI
OFFICE word processor showing the
various printout options available.

This is a demonstration of the MINI OFFICE word processor
showing the various printout options available.

This is a demonstration of the MINI OFFICE word processor showing the
various printout options available.

Page 1

This is a demonstration of the MINI OFFICE word processor showing the
various printout options available.



TOTAL SPENT



Please send me a copy of Mini Office for my Amstrad CPC464

☐ I enclose cheque for £5.95 made payable to
Database Publications Ltd.

I wish to pay by

☐ Access ☐ Visa No. _____ Expiry date _____

Signed _____

Name _____

Address _____

POST TO: Mini-Office Offer, Database Software,
68 Chester Road, Hazel Grove, Stockport SK7 5NY.

A3

the unbelievable
£5.95
CASSETTE

DATABASE SOFTWARE

Grind the Boston Bangers into dust

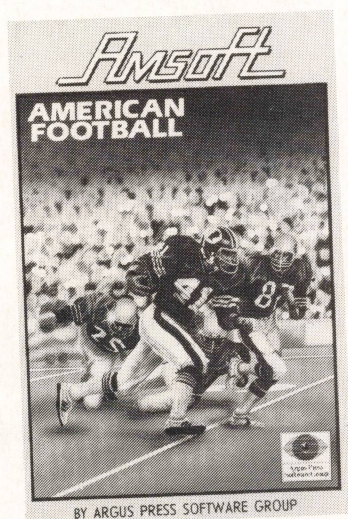
THERE is a growing number of people in the country who spend their Sunday evenings glued to the TV watching the Kentucky Chickens grinding the Boston Bangers into the dust – or vice versa.

Until now I had not been part of this group – I only watched the program to see the cheer leaders. However since I received **American Football** from APS under the Amsoft label, the mysteries of “first down and 10 on the 20” now have a new meaning. They didn’t have one at all before...

The game can be played against the micro or another player.

It isn’t the type of program in which you control a player and actually play the game – manual dexterity is not called upon at all.

In this game you are the coach, the eventual outcome of the match depending entirely on the team tactics



you employ. There is a comprehensive booklet which accompanies the program. An

initial section covers the rules, tactics and terminology involved while the rest describes how these are related to the computer version.

Each team has a collection of set manoeuvres – plays – at its disposal. Of the hundreds available in the real game, you are provided with a selection of four defensive and ten offensive plays.

As coach you are asked to enter the play you think would be most suited to the present situation. Having made your selection you are informed of the opposition’s tactics.

It is then a matter of sitting back and watching the two miniature teams engage battle. The position of the ball is known at all times as the

player in possession flashes.

You are informed in the instructions that the micro decides upon the play using a knowledge of the game and by analysing your previous plays. It doesn’t just wait and see what your selection is and then make its own. That’s what they all say.

American Football should appeal to both novice and expert alike. The graphics are simple but effective, and the tactical skill to luck ratio is just right.

Weighing in at eight stone wet through I think that this is about as close as I will ever get to playing the real game.

Jon Revis

Memories of WG..

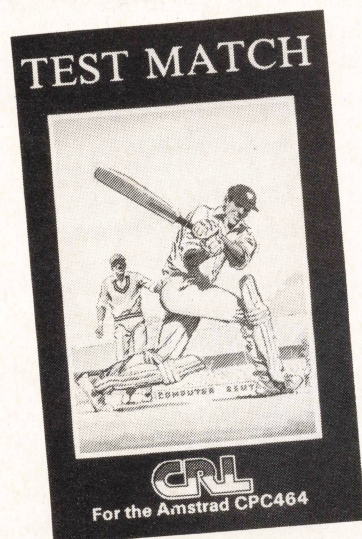
NOT being a cricket fan I was a bit dubious when handed **Test Match** by CRL to review. However there was a pleasant surprise in store.

In this Amstrad version of Limited Overs Cricket, you captain one of the teams while your opponent steers the other. There is no you v micro option.

The program features animated fielders, bowlers and batsmen together with a full scoreboard which is constantly updated as the game progresses. It covers a variety of events from a 40 over John Player League fixture to a lengthy 99 over match for the CRL Trophy.

You can either use players in the micro’s memory or – better yet – dredge the past to nominate a real team of experts. Does the name of W.G. Grace come to mind?

Both batsmen and bowlers are graded, and a bowler’s effectiveness is reduced for every five overs he bowls. Let’s



face it, electronic sportsmen tire too.

It would take up far too much space to reiterate how the program decides a player’s fate – suffice it to say that the preliminary instructions go into this in great detail and seem pretty logical.

After sorting out who is

playing whom the micro flips a coin and the winner opts to bat or field. I went for the John Player League game – 40 overs a side with an eight over limit per bowler.

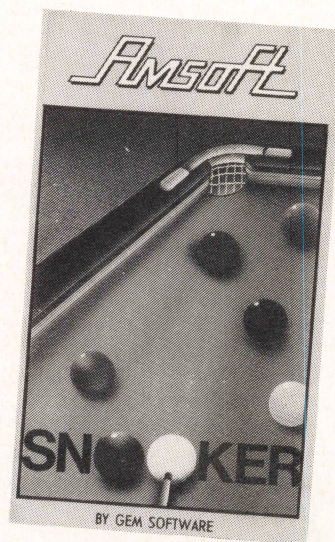
The screen presents a bird’s eye view of the field. Away goes the bowler, and if the batsman gives a polished performance and clouts the ball resoundingly the screen asks whether or not the batsmen should run. It takes a bit of practice here before you can judge whether or not to chance your arm – or legs.

Then it’s back to the scoreboard again and the choice of a new bowler – the micro refuses to let you use the same one in the next over.

The graphics, and colour, are quite good although movement is on the jerky side. But the sense of participation is really there, thanks largely to the run-or-not-run option.

Umpire’s verdict: Nice one CRL. Cricketers will love it.

Jed Glover



Eat your heart out!

SNOOKER is a game that has grown in popularity on a massive scale over the last few years, reflected in the large number of snooker games available for almost every micro.

In **Snooker**, from Gem under the Amsoft label, the user is first presented with a

series of options. Initially there is a choice of six, 10 or 15 red balls to determine the length of the game.

Should you be all alone then you can still enjoy a quick frame by selecting the practice option.

Finally you can play the game using numbered colours, allowing green screen Amstrad users to play without too much difficulty.

Having played snooker on several other micros I have often found difficulty in distinguishing between the different colours. This did not prove a problem on the Amstrad owing to the many colours available.

The game can be played using either joystick or keyboard. Using the cursor key cluster you are able to determine the direction of travel of the cue ball, the speed of the shot, and also to apply a combination of top, bottom, left and right spins.

With such a wide variety of options it is very easy to play real snooker shots, leaving the cue ball exactly where you want it after potting a red.

This is assuming you know how it is done in the real game!

Another similarity with the real game is that it is quite difficult to pot a ball. It is not one of those games where a near miss will suddenly vanish down the pocket – if you want to pot a ball then you have to be spot on.

Should you play a similar game of snooker to me, then this may be an advantage, as I tend to pot more whites than reds!

The mechanics of the game were acceptable – the balls moved quite slowly but the angles resulting from collisions and contact with cushions were very accurate. The latter is much more important than the speed of play.

Overall I found the game to be very professional and the level of ball control provided allowed the budding hustler to show his paces.

During yesterday's evening session I notched up a break of 15. Steve Davis, eat your heart out.

Jon Revis

3-D brain teaser

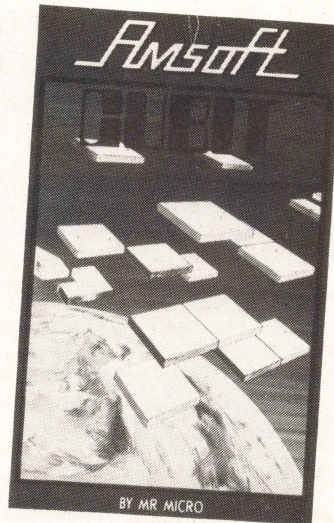
THE word addictive is bandied about in software reviews like a foaming pint at a drunkard's convention, so I'll not use it in relation to Amsoft's **Cubit**.

But I should – not because Cubit is enticing, demanding, coaxing and all the rest of the appropriate adjectives, but because it infuriates me.

Let's face it, this game is just an extension of your old noughts and crosses. A few minor embellishments – like four squares instead of three and another three squares stacked on top of the original ones – make it a three dimensional brain teaser.

Now if you use your loaf and pick the right square on which to start with noughts and crosses you just can't lose. With Cubit, playing against the micro, you just can't win. At least not for a long, long time.

Amsoft has done the decent thing and allowed you to play a human adversary instead of the Amstrad, and in this case the odds are more or less even.



But playing the CPC464 I'm on to a loser every time. Which is why the game is infuriatingly, compulsively unput-downable.

Peter Gee

Destined for fame

SNOWBALL from Level 9 is destined to be a classic.

You play the part of Kim Kimberley, secret agent, on

board a starship en route to Eridani A.

Along with 180,000 other colonists, you are hibernating inside the ammonia ice shell that serves as protection for the colonists and fuel for the main drives. Your mission is to guard the ship from sabotage.

When your freezer-coffin awakens you with the Snowball still in flight, you know that something is very wrong. Your task is to fulfil your mission by seeking out the saboteur.

Your first problem is getting out of your coffin. You are confused and disorientated by the long hibernation and it takes a few minutes to



Soon disappointed

I MUST admit to being quite pleased when I came across **Amstrad Basic: A tutorial guide Part 1**.

With its ring binder format, cardboard case and the two cassettes, it looked very promising, and surely anything would be better than the User Instructions.

So it was with a pleasant feeling of anticipation that I volunteered to review it. Sadly I was soon disappointed.

The first indication of what I was letting myself in for was when the text asked me to put First steps in Basic – Data-cassette A into the datacorder. There was no Datacassette A.

However, I guessed that the cassette marked Basic – A Tutorial Guide Side 1 was the one required and I was right. Still it hardly gave me confidence.

And when I was told that

the first program on the tape was the hello program when in fact it was a header, that didn't help.

Still that was just the tape, it could have been that the text was better. Sadly it wasn't. It lurched from keyword to keyword with no apparent method or purpose. The explanations were poor or totally lacking – what is a graphics cursor? – and the lack of diagrams was a big drawback.

The programs with the text weren't much use either. They were hardly inspiring and seemed to reflect the haphazard manner of the text.

However, all this pales into insignificance compared with the self assessment tests that come on the second tape. To my mind they are a complete waste of time, contributing little or nothing to the text.

It seems pointless having

them on a tape when they could just as easily be in the main text and the cassette used for some hopefully more illustrative programs.

It's not all bad news though – there are two good points to be mentioned. The first is the chapter on program design which stands head and shoulders above the rest of the text.

The second is Dave Atherton's Buzzword Generator, which bears more than a passing resemblance to the excellent Buzzword Generator by Mike Cook published on Page 22 of the January 1984 issue of *Electron User*.

A pity that Mr Atherton couldn't have captured more of Mike Cook's almost inimitable style.

Apart from these two things I cannot recommend this tutorial guide.

Nigel Peters ▶

discover the hatch lever.

Your second problem is likely to be the Nightingales – robots fitted with hypodermics. Their purpose is to ensure that any colonists who wake are injected with a drug that puts them back to sleep.

Unfortunately for you, the saboteur has anticipated your awakening and substituted his own drug – a deadly poison. A few lives later, you should reach the relative safety of the floor above.

Your quest for the saboteur now starts in earnest and you must search for a means of leaving the colony part of the ship in order to reach the main control room. To do this you have to go outside the ship, so try to wear a spacesuit.

To reach the main controls you need to ascend a cable network called Jacob's Ladder. Having done this you are now ready for the final showdown with the saboteur.

Level 9 have long been known for producing excellent adventures. Snowball, with its 7,000 (yes!) locations and 200 word vocabulary maintains that tradition. The program is totally logical, though it rarely seems so at the time.

The plot is well constructed, atmospheric and full of fiendish puzzles that will keep you awake until the early hours.

I would put it at about above-average difficulty, which means that beginners should get some practice in first.

Overall, an excellent adventure that is a joy to play. Highly recommended.

Merlin

Venture into the beautiful property world

If you've ever fancied trying to make money out of rented accommodation without running the risk of losing a penny, then **Country Cottages** from Sterling Software will appeal to you.

It is a two player management/strategy game which bases its theme on the purchase and leasing of property – namely cottages.

You start the game penniless and must specify the target amount that will decide the winner. On this amount – £20,000 to £100,000 – will depend the length of the game.

There are nine levels of difficulty and these decide the seriousness of any of a number of hazards – storms, burglars, fires and ghosts. On these can rest the future of any property you own.

The object is to purchase offered accommodation by bidding for it at an auction. To do this the bank will lend you for starters up to £100,000. This amount increases later as you become more affluent.

The property is offered, described in detail, and a minimum price is set before the bidding starts. Naturally the highest bidder becomes the new owner.

Selecting the option to look

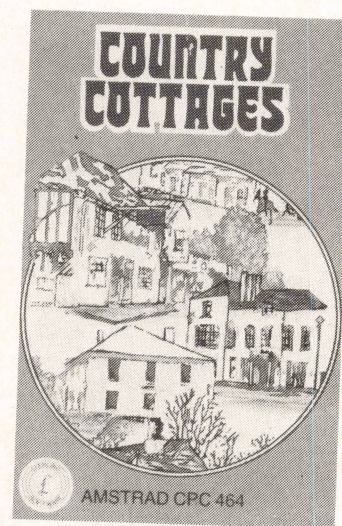
at the property will reveal the program's superb graphics. Sterling introduces it as 'Landscape Creation, a revolutionary concept in computer graphics giving an infinite number of different scenic views'.

It is certainly very impressive as you enjoy a new series of pictures every time the game is played. The scenery reflects the passing months of the year with snow-capped hills, valleys carpeted with spring flowers, and a bright summer sun shining over a still azure lake.

Sounds too good to be true? If you don't want to play you could just sit and watch the beautiful pictures instead. Sterling have even catered for this option in one special key.

When you advertise for tenants you meet a multitude of weird and wonderful characters such as Messrs Musbelly, Wagslim, Jibjaws and Mopwinkle, to name but a few.

You set the rent and your potential tenant either accepts or refuses depending on your greed or otherwise. As the



game progresses you are automatically presented with a monthly balance sheet which shows interest paid or received depending on the state of your account.

You are also shown an account of your present status with full descriptions of all properties, rents and tenants. This feature can be recalled at any time using the inspect option.

The hazards play a great part in the game and add quite a lot of humour. Storms are depicted quite dramatically, as are fires with the approaching engines, and the ghostly presence which frightens away your tenants is something to be heard.

I played this game with my 17 year old son who is 'invader mad' and after a while he admitted that he quite enjoyed it.

I found it fun and I'm not particularly keen on management games. Those who are would enjoy this even if there were no graphics, but these certainly add that final sparkling professional touch.

I think this game has got that certain something that could make it very popular with people of any age.

Alan McLachlan

Simple, but be warned!

HARRIER Attack from Durrell Software is not new – the storyline is simple.

You take off from the deck of an aircraft carrier situated off an unknown island (Malvinas?) and fly over a barren landscape – if you can avoid the missiles.

The intercepting jets and flak from ground fire try to prevent you reaching the township. After a period of bombing you can return to the carrier and hopefully make a

successful landing.

The screen display gives a fast moving outline of the terrain and indicators of speed, fuel and so on.

However, the use of the space bar on the bombing run feels a little clumsy.

An additional feature that also scores extra points is a last second eject option – the Escape key.

The aircraft is equipped with missiles and bombs for the pilot to hit the land based

weapons, the patrol boat and the enemy aircraft.

But be warned – the missiles move up and down after they have been fired if the aircraft is moved up or down.

The option of five skill levels helps the novice acquire the art of the end game – landing the plane – without having to have the immense dexterity necessary to fly over the island on level five, which is strictly for veterans.

John Woollard

THE first thing I noticed about **Xanagrams** was that it was suffering from schizophrenia – or at least, very mixed parentage.

From the Amsoft stable, the cover describes it as by Postern, yet its copyright is Dean Software.

Still, a program that's a cross between an anagram puzzle and a crossword is supposed to be a bit baffling, isn't it?

The idea is that you have to sort out a set of jumbled letters into the word they came from. The letters appear on the left of the screen and you must slot them into the spaces provided in the right order, recreating the original word.

You position the letters by moving a cursor via the arrow keys. Once you've arrived at the appropriate space, you simply type the letter you think should go there and wait to be enlightened.

If you were correct the letter moves into position and you score 30 points. If not, it beeps, and you lose five points – as I soon found, the program caters for negative scores.

A worthy attempt



If you really need a clue, pressing 1 will reveal the letter under the cursor – but it will cost you points, particularly if it's the first letter of the word.

If you've forgotten what the keys do, pressing Enter conveniently summarises them for you, while if the sound effects are driving you wild, pressing 5 will eliminate them.

To surrender, press 3 and all will be revealed.

You don't have just single words though. There's an option to allow you to work on mega-anagrams of up to five words. The spaces for the words then form an interlocking grid rather like crossword lights. The letters of the constituent words are presented as one gigantic anagram.

And you can have your anagrams presented to you at three levels of difficulty – junior school child, senior school child and adult.

What I found irritating was that after each game, even if only a single word were chosen, both options were prompted for again. Surely you should default to the last option you chose and be given a key for altering this when you wish?

As it stands the game is enjoyable enough. However

with a little effort if could have been even better. For instance, when you're moving around the grid the cursor doesn't auto-repeat, which can be tedious. And an optional time limit would have been fun.

Educationally though, it shows its limits even more. To be useful the vocabulary of the three levels would have to be far more differentiated than it is.

Of course teachers are notoriously difficult to please, but an option of loading your own vocabulary would have solved this – and made the program of great benefit to teachers.

All in all a worthy attempt, and something that word-puzzlers will enjoy for a while.

However, as a crossword addict myself, I soon found the straight diet of anagrams rather tedious and reached for my copy of the Grauniad...

Mike Bibby

Mr Punch turns villain!

IN my childhood the hero of the Punch and Judy show was always that long-nosed, big-chinned tyrant who was for ever chanting 'that's the way to do it' at the top of his high pitched unforgettable voice.

Nowadays the hero is the policeman, or at least that would appear to be the case if Amsoft's **Punchy** is anything to go by.

The story goes that Mr Punch has locked Judy away in the booth and she has called on the brave bobby to rescue her.

You control the bobby's movement across the booth and can also make him duck and jump when necessary. And believe me, it will be necessary when you see some of the obstacles.

You start off at the left hand side of the stage and your target is the right where you can see the baby's pram. If you can reach this you will hear a



bell ring and move on to the next screen.

Before you do you must sit through eight bars of "rock-a-bye baby" – unless you discover you can rapidly move on with a quick press of the Copy key.

As you head for your target, a variety of rotten tomatoes

and custard pies are hurled at you from the wings on both sides of the screen.

You must duck or jump these depending on their height which varies with the screen. If you are hit, one of your three lives is lost and you must start again on the left of the same screen.

Once through screen one things get a little more difficult. The stage has the appearance of battlements which the bobby must jump as well as avoiding the missiles.

Sounds familiar? The game is in fact a clever variation of the Hunchback theme, and extremely well done.

As you get further into the game you are met by a pit full of alligators which you can only cross using a constantly moving magic carpet.

The villainous Mr Punch appears tucked away in the next lot of battlements trying to stab your most intimate

areas with his sword. You can jump over him but it's easy to forget while doing so the custard pies and tomatoes which are still being hurled at you with gay abandon. All rather hair raising.

Occasionally Judy throws sausages which the bobby can catch. The idea is to try for a total of three which will transport him – at the press of a key – immediately to the next screen. This is a useful facility recommended for dealing with some of the more difficult screens later.

There are in fact 16 different screen formats. These are repeated ad infinitum for the whiz kid who finds himself able to handle them. I only managed the first five, but I did see my young son reaching later screens.

The game is fun to play and is quite addictive.

John Myers

Can do better...

IN **Atom Smasher** by Romik under the Amsoft label, you pilot a miniaturised craft into the heart of a runaway nuclear reactor. Here your task is to dispose of the protons in order to delay melt down.

Your ship is a simple triangle, typical of Asteroids games. This you can rotate left or right then thrust to the desired position where you can fire at the target. Easy, no doubt, if you are used to it but infuriating if you're not.

When you move, fuel is used but this is replenished when you are stationary. An important point this as the game automatically ceases once you run out.

Hitting the proton is a task and a half. It moves around the nucleus so erratically that to track it proves virtually impossible to my fuddled brain. I had great difficulty controlling the craft and it seemed easier to sit, wait and hope, then take the occasional pot shot.

However this proved not to be too successful because of the various hindrances intended to make life difficult.

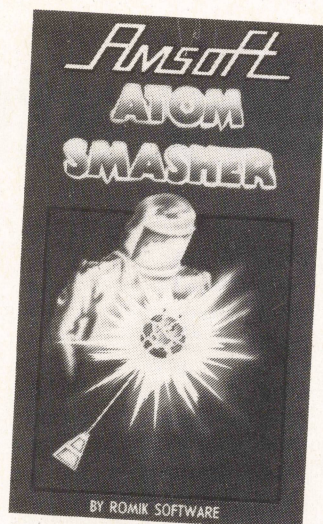
Floating around the nucleus at the start is an electron which makes life positively hairy as it cannot be destroyed by your laser, and a collision with it results in instant disaster. Fortunately you only lose one of your lives, recovering immediately to carry on the fray.

Each time you manage to shoot a proton another electron is released to add to your problems and the impending melt down is speeded up.

The temperature of your laser rises the more you fire it so it pays to be fairly frugal with the old ammo. Overheating will also result in an instant end to the game.

As the temperature rises in the reactor more debris is created as melt down approaches. You must control this by zapping it but this creates heat which creates debris which you can zap which creates heat and creates debris and so on...

The graphics are quite good



and excellent use is made of sound.

The game can be played using the cursor keys which I can't handle. Fortunately there is a facility to define your own keys and this made life just about bearable. There is also a joystick option, a freeze game facility and nine levels of difficulty.

I'm normally fairly tolerant with zap 'em shoot 'em games but I must be honest and admit that this game bored me to tears as I regularly made very little progress even at the easiest level.

My lack of success was probably my own fault and I suspect the game will appeal to the younger element, but with the quality of software around these days it has a lot of competition. I'm sure Romik can do better.

Hilda Pratt

Life will never be the same

EVERYTHING in the garden was rosy. Frank was happily trotting to pick up a ripe banana when *they* appeared out of nowhere. From that moment on, life would never be the same.

This is the scenario of **Fruity Frank** – the latest arcade offering from Kuma. You guide the little fellow round the screen collecting fruit while doing your best to avoid the various meanies intent on seeing you off.

The first meanies are Edward Heath look-alikes ably supported by a damson that suddenly changes into a purple people-eater with gnashing white teeth.

These are closely followed by the green letter monster and later, an evil creature resembling a strawberry.

It would seem the screen is an orchard, but one in which Frank can tunnel downwards in his search for the forbidden fruits.

These evil critters can be disposed of in two different ways. The first is to throw the ball you are carrying at the start of the game. This travels along the tunnels bouncing off the walls until it hits something.

The other option is to use the apples placed in various



positions around the screen. Place one over a vertical path – either by digging out from underneath it or by pushing it over one – and it will fall, crushing anything in its path.

The game has acceptable key choices and supports joysticks.

The graphics, in true Kuma tradition, are superb and excellent use is made of sound.

My only criticism is the lack of instructions. They are non-existent on the cassette insert and the only ones on-screen are the key locations.

Keith Chesworth

Escape from the pit

THE graphics of Amsoft's **Roland in the Caves** are excellent – it looks as if you are at the bottom of a deep pit surrounded by flesh eating plants with a deadly pterodactyl flying about.

Your adventure starts in the year 2464 AD. Exploring the strange planet of Ivorus you voluntarily mutate into the same form as the aliens – small fleas.

The object is to escape from the pit by using your flea-like powers to leap from ledge to ledge.

On the walls of the cave are flesh eating plants and if the

pterodactyl doesn't get you, they might.

The keyboard control is good. The distance and height of your jump is proportional to the time the jump keys are pressed down. This increases the necessary skill.

The twist to the problem is that if you eventually get to the top you fall in again. This time the flesh eating plants have multiplied and they are determined to prevent you leaving again.

Good luck if you buy this game – you will certainly need it.

John Woollard

REVIEWED SO FAR

American Football	APS
Amstradraw	B.S. McAlley
Astro Attack	Amssoft
Amstrad Tutorial	Amssoft
Atomsmasher	Romik
Blogger	Alligata
Country Cottages	Sterling
Cubit	Mr Micro
'Er*bert	Microbyte
Flight Path 737	Anirog
Football Manager	Addictive
Fruity Frank	Kuma
Forest at World's End	Interceptor
Galaxia	Kuma
Gems of Stradus	Kuma
Ghouls	Micro Power
Happy Letters	B.E.S.
Harrier Attack	Durrell
Happy Writing	B.E.S.
Hunchback	Ocean
Map Rally	B.E.S.
Manic Miner	Software Projects
Message from Andromeda	Interceptor
Punchy	Mr Micro
Roland in the Caves	Amssoft
Royal Quest	Timeslip
Snooker	CDS
Snooker	Gem
Snowball	Level 9
Test Match	CRL
The Moors Challenge	Timeslip
Timeman One	B.E.S.
Timeman Two	B.E.S.
Wordhand	B.E.S.
Wordwise	B.E.S.
Xanagrams	Postern



National Micro

Look at
the great
free gifts
you get
when you
buy from
nmc

FREE +
Pack of 12
software titles
with every CPC464
WORTH £108



ORDER FORM

Post to:
NATIONAL MICRO CENTRES,
36 St. Petersgate,
Stockport SK1 1HL

Item Please supply the following: Qty £ Total p

Attractive credit terms
Phone for details

Please indicate method of payment:

- ☐ Cheque payable to
National Micro Centres
☐ Access/Barclaycard No.

Name
Address
Tel. No.
Signed CA 3

Carriage
TOTAL

Please allow up to 28 days for delivery

AS SEEN
ON TV!

AMSTRAD
CPC464

Centres

Call in for a demonstration from our friendly sales staff at:

National Micro Centres
36-38 St. Petersgate
Stockport SK1 1HL
 or 'phone on:
061-429 8080/8074

FREE

Subscription to
 Computing with
 the Amstrad
WORTH £12



FREE

Amstrad lead with every
 MT80 Dot Matrix printer
WORTH £25

£249

Mail orders:
 Carriage £7



**All prices
 include VAT**

We also stock a wide range of other computers:

BBC "B"	£389
BBC "B" + DFS	£469
ELECTRON	£189 + 5 Free Games
COMMODORE 64	£189
SPECTRUM	£124 + 6 Free Games
SPECTRUM PLUS	£174 + 6 Free Games

Carriage on computers £7

**We also stock a wide range
 of software for ALL micros**

**Official Amstrad
 Joystick**

only £14.95

Mail orders: Carriage £1



£359 **£249**

Colour

Mail orders: Carriage £7

Amstrad Analysis

HAVE you ever wanted to display a set of numbers with all the decimal points aligned under each other? That's the task set this month. Centre Point solves the problem using the *LEN* and *MID\$* commands to dissect the numbers and find where the decimal point occurs.

Get right to the point with Trevor Roberts

Line number			
10,20	Tell humans the title of the program and who wrote it. The Amstrad itself ignores everything after the REM.	90	Prints out <i>number\$</i> each time round the loop cycle.
30	Puts the micro into Mode 2, the 80 column mode.	110	Stops the program from crashing into the subroutine below.
40-100	Form a FOR . . . NEXT loop which cycles five times to deal with each of the five numbers in turn.	120-180	Form the subroutine that searches <i>number\$</i> for its decimal point.
50	Each time round the loop reads the next number from the data list and stores it in the string variable <i>number\$</i> .	130-170	Make up a FOR . . . NEXT loop which cycles once for each character in <i>number\$</i> .
60	Ensures that <i>offset</i> is set to zero each time round the loop.	140	Takes one character from <i>number\$</i> and stores it in <i>check\$</i> . By the time the loop has finished, every character in <i>number\$</i> will have been stored in <i>check\$</i> .
70	Calls the subroutine that locates the position of the decimal point – if any – in <i>number\$</i> .	150	If <i>check\$</i> is a decimal point then its position – given by the value of <i>search</i> – is stored in <i>offset</i> .
80	Ensures that all the decimal points are printed under each other. This is done by making the print position 40 minus the number of figures left of the decimal point in <i>number\$</i> .	160	If the whole string has been checked and <i>offset</i> is still zero then the number has no decimal point. One is added to <i>offset</i> to give the position of the invisible decimal point at the end of a whole number.
		180	Ends the subroutine.
		200	Holds the five numbers to be printed.

```

10 REM CENTRE POINT
20 REM Trevor Roberts
30 MODE 2
40 FOR loop= 1 TO 5
50 READ number$
60 offset=0
70 GOSUB 130
80 LOCATE 40-offset,loop
90 PRINT number$
100 NEXT loop
110 END
120 REM SEACH FOR DECIMAL POINT
130 FOR search=1 TO LEN(number$)
140 check$=MID$(number$,search,1)
150 IF check$="." THEN offset=search
160 IF search=LEN(number$) AND offset
   =0 THEN offset= LEN(number$)+1
170 NEXT search
180 RETURN
190 REM THE NUMBERS TO BE CENTRED
200 DATA 1.2,12.3,12.34,123.45,123.456

```

FOR...NEXT loop reads *number\$* and calls subroutine

Subroutine to find decimal point position

Stores number from Data list in *number\$*

Zeroes offset each time round loop

FOR...NEXT loop sifts through *number\$* with *MID\$*

Data list

This is the end product – all lined up

1.2
12.3
12.34
123.45
123.456

Now you've started computing with the Amstrad
you'll want to read **EVERY** issue of...

Computing *with the* **AMSTRAD**

It's the only way to keep right up to date with what's
going on in the ever-expanding world of the Amstrad.

SPECIAL OFFERS!

How to protect your CPC464

Protect your micro with our luxury
dust cover made of soft, pliable,
clear and water-resistant vinyl,
bound with strong cotton and
decorated with the magazine's
logo.

DUST COVER ONLY £3.95

How to keep your collection complete

Bound in rich burgundy pvc and
bearing the Computing with the
Amstrad logo, this handsome
binder will hold a year's supply of
the magazines, firmly secured in
place with metal rods.

BINDER ONLY £3.95

You can also use the form below
to order back issues of
Computing with the Amstrad.

Look what you will get each month:

- ★ Reviews of all the very latest games,
educational and business programs now
being produced for the Amstrad.
- ★ Lots of listings you will be able to key in
yourself – games, utilities, graphics –
covering the whole field of Amstrad
computing.
- ★ In-depth independent evaluations of all the
new hardware add-ons now being
developed to make your Amstrad much
more powerful and much more versatile.
- ★ Lots of easy-to-follow features on everything
to do with the Amstrad. Whether you're a
beginner or an expert, you'll always find
something to delight and intrigue you.

We're making *Computing with the Amstrad*
into the most exciting computer magazine
ever – so don't miss an issue! If you've got
a CPC464, or about to get one, let
our experts show you how to
make the most of this really
versatile computer!

Your last chance to save £2 with our introductory offer!

Take out a 12-month subscription now
and pay only £10 instead of the
normal £12.

This special offer applies to UK
readers only and is valid until
March 31, 1985.

Please send me:

- ☐ The next 12 issues of *Computing with the Amstrad*
at the special introductory offer of £10.
- ☐ January's launch issue at £1.
- ☐ Last month's issue (February) at £1.
- ☐ Dust cover at £3.95.

I wish to pay by

☐ Cheque

Card No.

Expiry date

☐ Access ☐ Visa

ORDER FORM

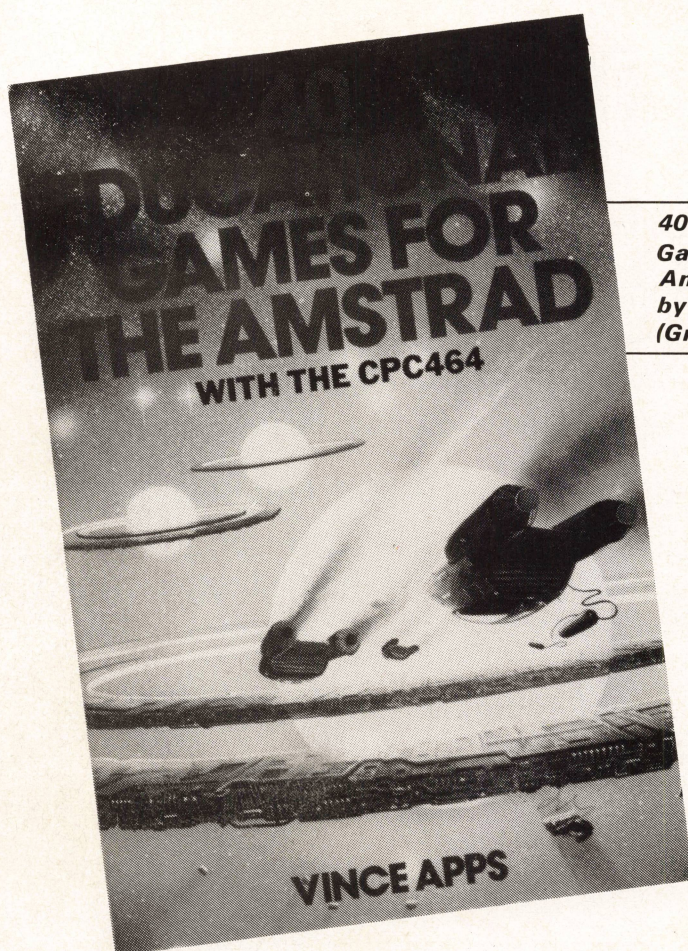
Name

Address

Signed

Send to: Database Publications,
FREEPOST, Europa House,
68 Chester Road, Hazel Grove,
Stockport SK7 5NY.

No stamp needed
if posted in
the UK.



40 Educational Games for the Amstrad, by Vince Apps (Granada £5.95)

Help for

IN a previous form this book was specific to the Electron, and the author has transferred the same games to the Amstrad – and that is where the problem lies.

The original programs were written to show some of the ways in which the Electron might be used by young people, and so the author worked within the constraints of the machine. For instance, the Electron's sound capabilities are fine but confined to one note at a time. The sound on the Amstrad is superb, yet little use is made of this facility.

However, perhaps this is only to be expected when a new machine comes out, as it is obviously easier to adapt than to write from scratch. Indeed, a lot of the books for the Electron were themselves originally written for the BBC Micro.

So what is this book like? Well, it is unashamedly aimed at younger users, and for them it is going to provide a lot of help. There are, for a start, 40

Here's a sample program, Parachute, from the book:

QUICK, quick, there is a plane crew parachuting down from a damaged airliner into a pine forest and the pilot's parachute hasn't opened. Only you can save him from crashing into the ground.

At the top of the screen you will see the height and the speed of the pilot's descent. To open the parachute you must divide the height by the speed to find out how long it will take for the pilot to land. If your answer is correct the parachute will open and your pilot will float gently to the ground.

Your crew has three lives between them so try and keep them alive as long as you can. The sooner you get the answer and type it in and press Enter the earlier the parachute will open and the more points you will receive.

You can make the game easier by reducing the numbers in lines 100 and 110.

```

10 REM parachute
15 REM(c) Granada Publishing
20 GOSUB 760
30 ENV 1,1,15,1,15,-1,7
40 HIGH =0
50 REM *****loop*****
60 MODE 0
70 BORDER 13
80 LIVES = 3:SCORE = 0
90 REM*****loop*****
100 SPED=INT(RND(1)*90)+10
110 ANS=INT(RND(1)*12+1)
120 HEIGHT = ANS*SPED
130 GOSUB 870
140 GOSUB 250
150 IF LIVES >0 THEN GOTO 90
160 MODE 1:BORDER 15
170 LOCATE 1,2:PRINT"PARACHUTE"
180 LOCATE 1,3:PRINT"*****"
190 LOCATE 1,5:PRINT"You scored ";SCORE
RE
200 IF SCORE >HIGH THEN HIGH = SCORE:
LOCATE 1,8:PRINT"a new high score !!!"
210 LOCATE 1,12:PRINT"Press a key to
play again ";
220 IF INKEY$<>"" THEN GOTO 220

230 IF INKEY$="" THEN GOTO 230
240 GOTO 50
250 REM*****
260 IN$=""
270 I=4
280 SOUND 1,1+I*2
290 LOCATE 11,I:PRINT" "
300 LOCATE 11,I+1:PRINT CHR$(228)
310 T=1
320 T=T+1
330 GT$=INKEY$
340 IF GT$="" THEN GOTO 400
350 IF GT$=CHR$(13) AND IN$<>"" THEN
GOTO 500
360 IF GT$=CHR$(127) AND LEN(IN$)>0 T
HEN IN$=LEFT$(IN$,LEN(IN$)-1):GOTO 39
0
370 IF GT$>"9" OR GT$<"0" THEN GOTO 4
00
380 IF LEN(GT$)<2 THEN IN$=IN$+GT$:FO
R L= 1 TO 50:NEXT L
390 LOCATE 18,5:PRINT IN$;" "
400 IF T<>50 THEN GOTO 320
410 I=I+1:IF I>21 THEN GOTO 280
420 LOCATE 11,21:PRINT" "
430 LOCATE 11,22:PRINT CHR$(231)
440 SOUND 1,0,150,0,1,0,7

```


the younger user

programs for less than £6, and this will give the beginner a gentle and inexpensive start to programming.

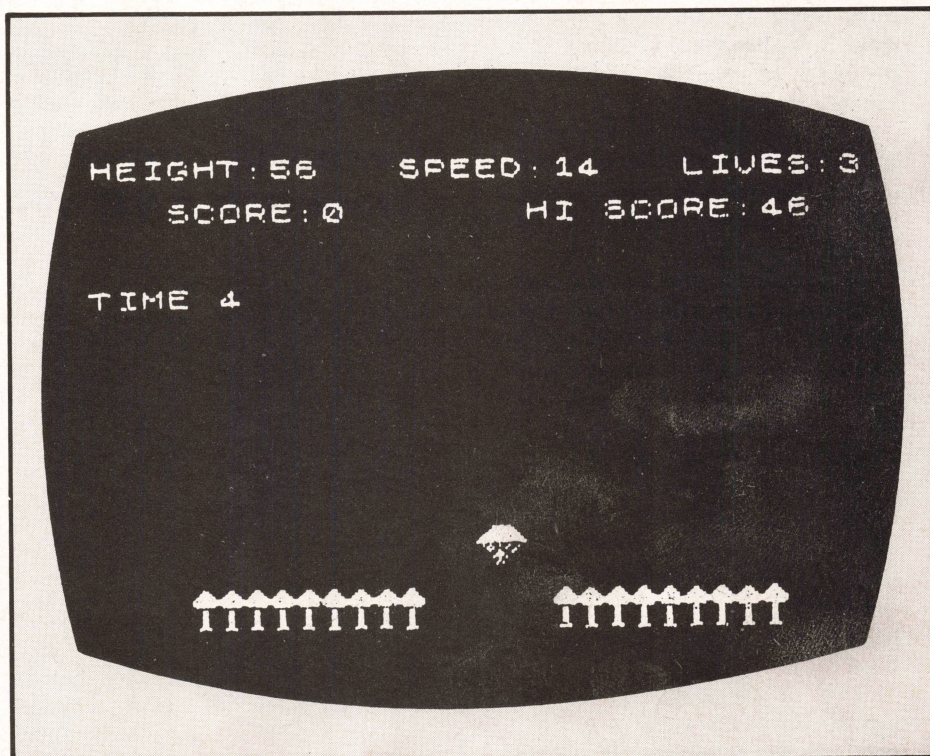
Some of the programs are quite good, and all have a slant towards enjoyment, so I am sure children will enjoy using them. The range covered is quite wide, from science to spelling and from the Highway Code to morse code.

Standards are there of course, but I was pleased to see a few newer ideas in this book. I cannot imagine that many children will find all the programs instantly useable, but there are program notes which might encourage the child to experiment.

These notes could be more expansive, and I would be happier to see more REM statements in the body of the listing.

Overall, I hope that better books will arrive in due course, but for the young and inexperienced user, this one represents fair value for money.

Phil Tayler



```

450 LIVES=LIVES-1
460 FOR L=1 TO 5000:NEXT L
470 RETURN
480 LOCATE 16,22:PRINT " "
490 RETURN
500 TRIES=TRIES+3
510 NUM=VAL(IN$)
520 IF NUM=ANS THEN GOTO 550
530 IN$=""
540 GOTO 390
550 REM
560 SCORE=SCORE+23-I-TRIES
570 FOR J= 2 TO 20
580 LOCATE 10,J-1:PRINT " "
590 LOCATE 10,J:PRINT " "
600 LOCATE 10,J+1:PRINT CHR$(224);CHR
$(225)
610 LOCATE 10,J+2:PRINT CHR$(226);CHR
$(227)
620 FOR K=1 TO 200:NEXT K
630 NEXT
640 LOCATE 1,12:PRINT"A successful la
nding"
650 LOCATE 4,15:PRINT"You score ";23-
I-TRIES;" points."
660 FOR K= 120 TO 240 STEP 10
670 SOUND 1,K,4

```

```

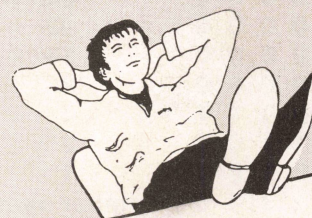
680 SOUND 2,K+1,4
690 NEXT K
700 FOR K=1 TO 2300:NEXT K
710 LOCATE 11,21:PRINT " "
720 LOCATE 11,22:PRINT " "
730 TRIES =0
740 RETURN
750 END
760 REM*****
770 SYMBOL AFTER 224
780 SYMBOL 224,0,7,15,31,63,127,255,0
1
790 SYMBOL 225,0,224,240,248,252,254,
255,18
800 SYMBOL 226,41,20,9,3,5,1,2,2
810 SYMBOL 227,36,72,16,128,64,0,120,
128
820 SYMBOL 228,0,0,16,56,84,16,40,40
830 SYMBOL 229,24,24,24,24,24,24,60,1
26
840 SYMBOL 230,0,24,60,126,255,255,12
6,24
850 SYMBOL 231,0,0,0,0,129,90,126,255
860 RETURN
870 REM*****
*
880 PEN 3

```

```

890 CLS
900 LOCATE 1,24:PRINT"HEIGHT";HEIGHT
910 LOCATE 12,24:PRINT"SPEED";SPED;
920 LOCATE 1,3:PRINT"SCORE:";SCORE
930 LOCATE 13,3:PRINT"HI:";HIGH
940 LOCATE 1,5:PRINT"LIVES:";LIVES
950 LOCATE 13,5:PRINT"TIME;"
960 FOR J=0 TO 19
970 IF J>7 AND J<14 THEN GOTO 1020
980 PEN 1
990 LOCATE J+1,21:PRINT CHR$(230)
1000 PEN 2
1010 LOCATE J+1,22:PRINT CHR$(229)
1020 NEXT J
1030 RETURN

```



Give your fingers a rest...

All the listings from this month's issue are available on cassette.

See our special offer on Page 19.

YOU'LL find this screen dump utility very useful. It allows you to make hard copies of any screen display in Mode 0 or 1, and it doesn't half help convince editors how good your arcade game really is – a picture is worth a thousand superlatives.

The program is capable of dumping both text and graphics, which means you can produce a dump out of anything you can create on the screen.

Applications aren't restricted to illustrating games of course. It would come in very handy producing graphs and so on for business programs. And you can have great fun dumping out simple drawings you have created.

But how can you do all this?

The first thing you need is a printer which is Epson compatible. This means something like a Kaga, Cannon PW 1080A and, of course, the Epsoms themselves. If you do not have such a printer then I'm afraid it's hard luck.

The printer has to be Epson compatible so you can directly program the print head to operate in what's known as bit image mode – this is selected by sending the printer a few control codes.

Bit image mode is required to allow graphic displays to be dumped because it will print any dot pattern sent to it. The process is not unlike defining a character such as a space invader using the SYMBOL command.

When data is received by the printer, it is treated as a binary bit pattern and is printed in a vertical strip eight pixels high. So if five items of data were sent – 1, 2, 3, 4, 5 – it would be printed as:

0 0 0 0 0	128
0 0 0 0 0	64
0 0 0 0 0	32
0 0 0 0 0	16
0 0 0 0 0	8
0 0 0 1 1	4
0 1 1 0 0	2
1 0 1 0 1	1
1 2 3 4 5 – number sent	

where "1" is a dot and "0" is a space. The least significant bit is at the bottom and the most significant at the top.

To dump the screen on the printer

Dump anything off the screen

with this screen dump utility
from ROLAND WADDILOVE

```

210 REM line feed is 1/36 th of
220 REM inch and sum=&4669
230 DATA CD,06,B9,3E,1B,CD,91,AA,3E
240 DATA 41,CD,91,AA,3E,04,CD,91,AA
250 DATA 21,8E,01,22,9B,AA,3E,09,CD
260 DATA 91,AA,3E,1B,CD,91,AA,3E,4B
270 DATA CD,91,AA,3E,40,CD,91,AA,3E
280 DATA 01,CD,91,AA,21,00,00,22,99
290 DATA AA,01,20,00,ED,43,9D,AA,48
300 DATA 2A,9B,AA,A7,ED,42,ED,5B,99
310 DATA AA,C5,CD,DF,BD,C1,FE,00,28
320 DATA 08,2A,9D,AA,7D,84,32,9E,AA
330 DATA 21,9D,AA,A7,CB,1E,0C,0C,79
340 DATA FE,08,20,D8,3A,9E,AA,CD,91
350 DATA AA,2A,99,AA,23,23,22,99,AA
360 DATA 01,80,02,A7,ED,42,20,BA,3E
370 DATA 0A,CD,91,AA,2A,9B,AA,01,08
380 DATA 00,A7,ED,42,22,9B,AA,30,88
390 DATA C9,47,CD,2B,BD,78,30,FA,C9
400 DATA 00,00,00,00,00,00,00
Ready
run
What is the paper colour ? 0
Ready
call&aa00

```

Screen dump of latter half of listing

all that is required is to scan the screen horizontally, looking at the pixels, converting them to a number with an identical binary pattern and sending that number to the printer.

Unfortunately, the Amstrad can only send codes 0 to 127 to the printer – this is because bit seven of each code is ignored.

We now have a problem. The top bit of each column will be ignored. This means that only seven eighths of the column can be printed.

The way round this is to do each row twice, the first time print the top four bits of the column and the second time printing the bottom four. This reduces the speed by half but seems to be the only way.

We achieve this by changing the line feed so that it only moves up half the normal distance. Now the remainder of the column can be printed.

A screen dump in Basic would be very slow. For this reason the program is written in machine code.

The dump routine is tucked away at the end of memory. To avoid any possible corruption by a Basic program, the MEMORY command has been used to limit the memory available to Basic. For those of you who are interested, the dump program starts at location &AA00.

Now it's time for the program. Type it in and save it.

It should be noted that some Epson compatible printers are slightly different. The difference that affects us is the distance moved each line feed.

On the Epson the minimum line feed is 1/72nd of an inch. On some printers the minimum line feed is only 1/36th of an inch. If your printer is the latter you will have to change the sixth number in line 240 to 02 – it is 04 in the listing.

The check sum should also be changed to &4669 – in line 180. Otherwise a splayed dump will be produced.

Running the program sets up the

machine code routine – this is stored as data statements. If you have mis-typed any data item the message ERROR will be displayed. Check through your program and change the data item in error.

When the data has been accepted you will be asked to enter the paper colour – in nearly all cases this will be 0.

This number is used to indicate to the dump program that the selected colour should not be printed. If the pen colour had been entered the dump would appear in reverse field – a negative of the screen.

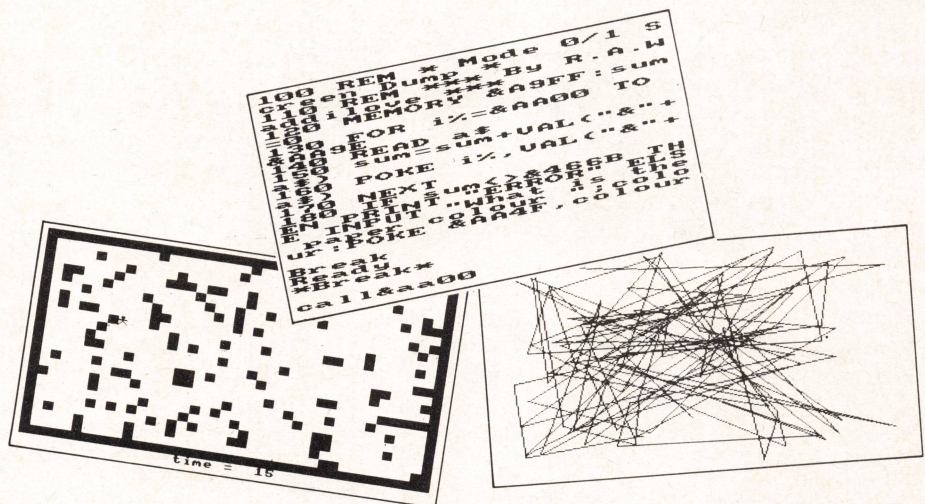
Once RUN, the dump routine is ready to be used. You can now load in the program you wish to dump.

At the required point in your program you should put a CALL &AA00 command which will cause the screen to be dumped when the line is executed.

The program will dump any Mode 0 or 1 screen, as long as MEMORY is not changed or the micro reset.

Each time you need to dump a program you should do the following:

1. Load in the dump program.
 2. RUN it.
 3. Load in the program you wish to dump.
 4. Put a CALL &AA00 command in the program where a screen dump is required.
 5. RUN the program.
- Happy dumping!



```

100 REM *** Mode 0/1 Screen Dump ***
110 REM ***** By R.A.Waddilove *****
120 MEMORY &A9FF:sum=0
130 FOR iZ=&AA00 TO &AA9E
140 READ a$
150 sum=sum+VAL("&"+a$)
160 POKE iZ,VAL("&"+a$)
170 NEXT
180 IF sum<>&466B THEN PRINT "ERROR"
    ELSE INPUT "What is the paper colour
";colour:POKE &AA4F,colour
190 REM ***** Machine Code *****
200 REM 15th number=02
210 REM line feed is 1/36 th of
220 REM inch and sum=&4669
230 DATA CD,06,B9,3E,1B,CD,91,AA,3E
240 DATA 41,CD,91,AA,3E,04,CD,91,AA
250 DATA 21,8E,01,22,9B,AA,3E,09,CD
260 DATA 91,AA,3E,1B,CD,91,AA,3E,4B
270 DATA CD,91,AA,3E,4B,CD,91,AA,3E
280 DATA 01,CD,91,AA,21,00,00,22,99
290 DATA AA,01,20,00,ED,43,9D,AA,4B
300 DATA 2A,9B,AA,A7,ED,42,ED,5B,99
310 DATA AA,C5,CD,DF,BD,C1,FE,00,2B
320 DATA 0B,2A,9D,AA,7D,84,32,9E,AA
330 DATA 21,9D,AA,A7,CB,1E,0C,0C,79
340 DATA FE,0B,20,DB,3A,9E,AA,CD,91
350 DATA AA,2A,99,AA,23,23,22,99,AA
360 DATA 01,80,02,A7,ED,42,20,BA,3E
370 DATA 0A,CD,91,AA,2A,9B,AA,01,0B
380 DATA 0B,A7,ED,42,22,9B,AA,30,8B
390 DATA C9,47,CD,2B,BD,7B,30,FA,C9
400 DATA 00,00,00,00,00,00,00
    
```

Full Basic listing

xZ=&AA99	CALL print	CP colour	SBC HL,BC
yZ=&AA9B	LD A,75	JR Z,next	JR NZ,loop2
aZ=&AA9E	CALL print	LD HL,(bZ)	LD A,10
bZ=&AA9D	LD A,64	LD A,L	CALL print
ENABLE ROM = &B906	CALL print	ADD A,H	LD HL,(yZ)
TEST = &BDDF	LD A,1	LD (aZ),A	LD BC,B
PRINT CHAR = &BDD2B	CALL print	LD HL,bZ	AND A
start CALL ENABLE ROM	LD HL,0	AND A	SBC HL,BC
LD A,27	LD (xZ),HL	RR (HL)	LD (yZ),HL
CALL print	LD BC,32	INC C	JR NC,loop1
LD A,65	LD (bZ),BC	INC C	RET
CALL print	LD C,B	LD A,C	LD B,A
LD A,4	LD HL,(yZ)	CP B	again CALL PRINT CHAR
CALL print	AND A	JR NZ,loop3	LD A,B
LD HL,39B	SBC HL,BC	LD A,(aZ)	JR NC,again
LD (yZ),HL	LD DE,(xZ)	CALL print	RET
loop1 LD A,9	PUSH BC	LD HL,(xZ)	LD (xZ),HL
CALL print	CALL TEST	INC HL	LD BC,640
LD A,27	POP BC	INC HL	AND A

Disassembly of the machine code

EDUCATIONAL FORUM

By
**GARETH
LITTLER**



WELCOME to the Amstrad Education Forum. In this new column we hope to keep you up-to-date with developments for the CPC464 in education as they occur.

The shortage of good educational software is starting to be rectified. A number of Sinclair Spectrum software houses are beginning to transfer their software to the CPC464 but we see suppliers for the Research Machines 380/480Z as a more important source of educational software for the Amstrad.

Johnathon Briggs of Logic Programming Associates discovered that when he downloaded Micro-Prolog for the RML 380Z to the Amstrad it would run without modification. This 18k assembler program will shortly be available.

Not all suppliers with software implemented on the 380 and 480Z will be so fortunate. They will have to make some modifications to their programs. Letters have gone to managing directors of software houses that offer primary, secondary and higher education programs for the 380 and 480Z suggesting that they should also offer these programs for the Amstrad.

Northern Computers are planning to offer multi-user licences for educational software which should allow a program to be used on any machine in use in one establishment. The availability of these licences will depend on the cooperation of those software houses who wish to take part.

The January/February 1985 "Computing in Print" booklet from Neat Quest now lists 13 books for the CPC464. We have also been impressed by the "Amstrad CPC464 Advanced User Guide" published by Sigma Press at £6.95.

The author, Mark Harrison, has included 40 program listings as well as a useful description of the basic commands, statements and functions of the system.

The first Amstrad User show will be held at Olympia between April 30 and May 2, 1985. Entrance should be free and admission restricted to over 16s only. The exhibition will be part of the British Electronics Week Show.

If you know of, or have good educational software, add-ons or books for the CPC464 then we would like to hear from you. We don't promise to review everything but we do want to use this Forum to provide reviews and news for the Amstrad's growing educational user base.

All correspondence should be addressed to: *Gareth Littler, Computing with the Amstrad, Europa House, 68 Chester Road, Hazel Grove, Stockport.*

* Gareth Littler specialises in educational software with Amstrad distributors Northern Computers.



PRIDE UTILITIES

3 great utilities for the CPC 464

ZEDIS A comprehensive yet friendly editor and disassembler, indispensable to the novice or Machine Code expert.

- * Full Z80 disassembly
- * Continual menu display
- * Easy to use, full error trapping
- * 3 windows and 80 column text
- * High speed Hex, code/string search
- * Forward/backward paging and scrolling
- * 3 O/P styles

- * Hex, code/string I/Ps
- * Full control of O/P to printer
- * Calc. relative displacement & jumps
- * Register inspection
- * Break point insertion
- * Full written instructions to dis. ROMs

CASSETTE £6.95 inc. P/P

RSX.SYCLONE A resident system extension which adds 3 new commands, offering 19 new facilities for your CPC 464.

- * Commands available from Basic
- * Load and List protected Basic programs
- * Copies all tested software
- * All file types catered for
- * Choice of 4 loading speeds, 1000 up to 4000 baud
- * Save your programs to Load in up to 1/4 of normal time
- * Convert existing software to high speed loading software
- * Full Error reporting
- * Antisyclone facility to add to your own programs
- * Full written instructions
- * List your 'Welcome Tape' easily.

CASSETTE £6.95 inc. P/P

PRINTER PAC 1 A resident system extension which can add 6 new commands, a total of 8 extra facilities to your CPC 464 when used with the DMP1 PRINTER or the EPSON RX80 and similar printers.

- * Specify horizontal position of dump
- * Specify which inks you wish to be considered for background
- * Dump current screen to printer
- * Copy the text only to printer
- * Abbreviated codes to printer. A great time saver
- * 3 new type styles for the DMP1 (not required for the Epson RX80).

CASSETTE £4.95 inc. P/P

*** Now available - Tape to Disc converter £7.95 incl P&P ***

S.A.E. for details of this great new program

Buy any 2 titles and get a 3rd cassette containing a Real Time Digital Alarm Clock, RSX Program, **FREE** from:

PRIDE UTILITIES (CWA),
7 Chalton Heights,
Chalton, Luton, Beds. LU4 9UF.

For Europe add
£1 per title

For Rest of World
add

SUNARO Software

BEST FOR AMSTRAD SOFTWARE

Order any two titles deduct £1 extra

Football Manager	6.95	Hunchback	7.85	Lords of Time	8.75
Flightpath 737	6.25	Gem of Stradus	7.85	Er Bert	5.25
Grand Prix Driver	6.25	Forth	21.95	Manic Miner	6.95
Fantasia Diamond	6.95	Happy Letters	7.85	Message from Andromeda	5.25
Snowball	8.75	Harrier Attack	7.85	Fruit Machine	7.85
Adventure Quest	8.75	Roland Ahoy	7.85	Tasprint 464	8.75
Colossal Adventure	8.75	Blogger	6.95	Sultans Maze	7.85
Erik the Viking	8.75	House of Usher	6.25	Concise Basic Manual	10.95
Forest at Worlds End	5.25	Steve Davis Snooker	6.95	Concise Firmware Manual	17.95
Jewels of Babylon	5.25	Fruity Frank	6.25	Happy Writing	7.85
Tasword 464	17.95	Dungeon Adventure	8.75	Roland in the Cave	7.85
				Roland on the Ropes	7.85

New titles available immediately upon release
Cheque/PO to:

SUNRO SOFTWARE (CA1)
PO Box 78, Macclesfield, Cheshire SK10 3PF

SHEKHANA COMPUTER SERVICES DISCOUNT AMSTRAD SOFTWARE

	Our RRP	Price		Our RRP	Price
Colossal Adventure	9.95	7.50	Jet Set Willy	7.95	5.95
Adventure Quest	9.95	7.50	Football Manager	7.95	5.95
Dungeon Adventure	9.95	7.50	Blogger	7.95	5.95
Snowball	9.95	7.50	Steve Davis Snooker	7.95	5.95
Return to Eden	9.95	7.50	Manic Miner	8.95	6.70
Lords of Time	9.95	7.50	Hunchback	8.95	6.70
Fruit Machine	8.95	6.70	Astro Attack	8.95	6.70
Grand Prix	8.95	6.70	Star Commander	8.95	6.70
Harrier Attack	8.95	6.70	Admiral Graff Spee	8.95	6.70
Codename Matt	8.95	6.70	Hunter Killer	8.95	6.70
Spannerman	8.95	6.70	Electro Freddy	8.95	6.70
Roland in the Cave	8.95	6.70	Flight Path	7.95	5.25
Roland on the Run	8.95	6.70	Jewels of Babylon	6.00	4.50
Roland on the Ropes	8.95	6.70	Message from Andromeda	6.00	4.50
Roland Ahoy	8.95	6.70	Pjaramama/Master Chess	12.95	9.99
Roland Goes Digging	8.95	6.70	Guide to Basic Part I	19.95	14.95

Overseas orders welcome. Please order stating:

(1) Program required (2) Amount enclosed (3) Name and address

Please make cheques & postal orders payable to:
SHEKHANA COMPUTER SERVICES
653 Green Lanes, London N8 0QY
Tel: 01-800 3156

Please send SAE
for further
titles available

All prices
include postage,
packing and VAT

WE already know how skilled and creative Amstrad users are, and we look forward to receiving your programs and articles for publication in future issues of *Computing with the Amstrad*. However before you send your masterpiece off to us there are one or two points that you ought to bear in mind to make all our lives easier. We call them the eighteen commandments...

The 18 commandments

A guide to presenting your masterpiece for publication

WHILE not wanting to put programmers' creativity into a straightjacket we've found that life can be made a lot easier for the magazine, our readers and the programmers themselves if we stick to certain standards.

It has also occurred to us that it's no good our just knowing what we want, we have to tell you, our potential contributors. So here are our 18 commandments. Don't be too daunted by the list – it's mostly just commonsense and good programming practice.

- Send us your programs on tape. There's no point in just sending a listing and asking if we're interested. You can't expect us to evaluate a program from merely reading a listing. We may be good, but we're not that good! A cassette with the program on is a must.

We don't use two part programs in the magazine. Games in two files may look professional but they're the kiss of death as far as the magazine is concerned. Too much can go wrong when people type them in.

- Avoid variable names that lead to confusion such as *1* and *I*, *O* and *o* and try to use lowercase variable names as it makes life easier for the reader who's trying to type it in and debug his errors. Meaningful variable names help as well – *aliens* is far more understandable than *al*.

- Tell us what the program is supposed to do and refer to it by name. You'd be amazed at the number of programs we get where the author forgets to tell us what it is all about.

In any subsequent correspondence, reference to "my program" can cause problems by its vagueness. Okay, we'd have the program on record somewhere, but life would be a lot easier all round if its author were

less modest and admitted he was the genius behind "Mega-invaders".

- Label everything with both the program's name and your own name and address. Keep your own copy of it, too. So far the only existing copy of one particular classic game hasn't disappeared in the post – but there's no reason to run the risk of yours being the first.

If it's a game let us know how to "cheat" so we can test out the higher levels. We're getting on a bit here and our reactions aren't as good as they used to be. (Not that they were up to much when they were as good as they used to be...)

And an adventure-type game or whatnot should come with a map of the rooms and any other crib sheet you possess. Much as we'd like to, we just don't have time to guess the name of Rumpelstiltskin's brother, no matter how much we admire your ingenuity. (Anyway he works in our artroom.)

- Put more than one copy of the program on your tape, recorded at different speeds and on different sides. And if you want the cassette back let us have a stamped addressed envelope with the name of the program on it.

You won't appreciate this unless you've run a computer magazine, but please send each different program on a different cassette. If not, we just can't handle them. The rule is, one program per cassette – though recorded several times on it.

- Let us have a printed listing if possible. Screen dumps or off-screen photos are much appreciated, though

not vital. Diagrams are always of use. Often a point that's difficult to put into words becomes clear as crystal when you sketch it out.

- Give a description of the program, what it does, why you wrote it, and outline the way it works and its variables and subroutines.

If it's a game let us have a plot. You'll get an idea of the sort of thing we want by reading the introductions to one or two of our games.

Maybe you could also give a few ideas for its improvement or expansion. Even if you can't get your upgrades to work there's a good chance that someone among our very talented readers will.

Every subroutine must be titled clearly with a REM and should be referred to by it. Again, make the title meaningful. Also when you GOSUB use a REM to indicate which subroutine you're using. For example:

100 GOSUB 1000: REM Move man

.

.

.

.

1000 REM ** Move Man ******

.

.

1100 RETURN

At first this may seem to be far too much fuss, but it's not just for the readers' benefit. As your programs grow you'll find that such REMs more than repay the effort by allowing you to keep track of your work.

When you write out your list of subroutines (vital) try to do it in the form:

100 example Shows how we want...

200 delay Holds things up...

where the line numbers refer to the lines where the subroutine is defined. Again, this helps by making things

Contributions should be sent to:
Features Editor, *Computing with the Amstrad*, Europa House, 68 Chester Road, Hazel Grove, Stockport SK7 5NY.

clearer to our readers – and you!

We don't expect your program descriptions to be classics of English literature, but it does help if they make sense and are easy to follow. Try reading them out loud – you'd be amazed how much such a simple technique can improve your writing.

Also if you get stuck trying to put something into words use this trick: tell someone what it is you're trying to put into words – then write it down. Before you reject this hint, try it – more than one professional writer owes his career to it.

It is good practice to renumber your program, starting at 10 in increments of 10 – the standard default. This way a missing line stands out like a sore thumb.

● Make sure that the program actually works. Try it out on your friends for their criticism (painful though it may be). The acid test is to ask them to type it in. And – when you find yourself muttering through clenched teeth, "How could anyone be that stupid?" (the answer is "regularly") – cast out the mote in your own eye and alter your program to take account of the feedback.

It's not easy to do, as the all-too-frequent blood feuds among the editorial staff here testify, but it's worth it.

Instructions can make or break a game. Make sure that yours really do instruct. They should be complete and it helps if the spelling and grammar are correct. Apart from causing confusion, such errors also make programs look amateurish.

As well as misspellings, bad grammar, split words and general untidiness are all to be avoided.

Following even the simplest program can cause problems for the most experienced programmer – don't add to them unnecessarily. One major cause of having to return programs for modifications is the colour/monochrome dilemma.

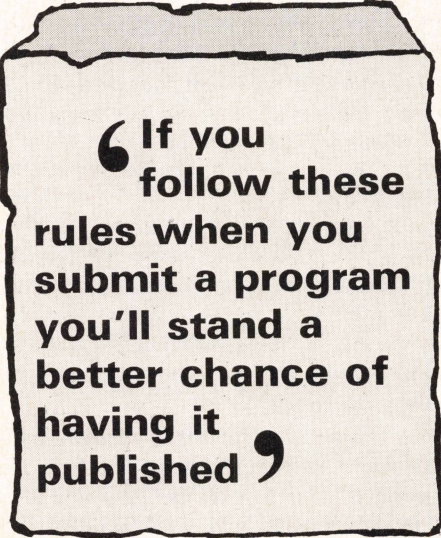
You can develop a beautiful program making use of all the splendid colour the Amstrad is capable of, only to find that the action disappears in an impenetrable fog on a monochrome monitor – and vice-versa.

If possible try your program on both versions of the monitor. User groups are invaluable here as they are in all aspects of program develop-

ment.

● Please do put lots of nice explanatory REMs in your programs. A couple of REM statements with nothing after them at the beginning of the program gives us room to put in our messages without messing up all the line numbers you have referred to in your program description.

● Avoid having just a line number with a colon and nothing else. It may make the program look neater but we won't welcome letters asking what the missing line was. Remember, people will be spending hours typing



'If you follow these rules when you submit a program you'll stand a better chance of having it published'

your programs into their micros. Make their life easier if you can.

● Double space all your written matter. This means leaving a blank line between each line of text – it's vital from our point of view. Try to follow our style. We have our own ways of doing things. We talk about modes in general but Mode 1 in particular. We press the Enter key, not the ENTER key as you might expect.

Just look how we do it in the magazine. Our programs are Program I, Program II, and so on; our diagrams Figure I, Figure II.

● If you must use long multiple lines don't go over about 175 characters, as people always add spaces when they type them in, then complain the lines are too long.

● Always put in the right number of NEXTs – don't just use NEXT followed by a comma, as it causes a lot of confusion.

Similarly with REM. You can abbreviate it with an apostrophe, but it's exceptionally easy to miss – for

the sake of typing two extra characters you can save our readers hours of frustration.

● Please, when you send us your work, include a separate page telling us that it is your own work, it has not been offered elsewhere and we have your permission to print it. If you don't, we'll have to return it.

● It's always nice if a program can have an alternative key or joystick option.

● One of the major causes of programs crashing is because the user inputs something the programmer wasn't expecting. All right, the idiot shouldn't type in –999 when you ask him his age, but believe me, they will, out of sheer perversity – particularly if the program is educational. There is something about CAL programs that brings out the devil in us all...

So try out all the unlikely options – if you don't, some poor user will.

Actually it takes a lot of skill to idiot proof a program, as it's delicately known in the trade.

Often you're so involved in getting the program to work as it's supposed to that you just can't make the mental leap needed to see it as the passively malevolent reader does. So try it out on your friends!

● Another irritation for a reader is when he sees something like:

PRINT" "

Exactly how many blanks is he supposed to enter?

Use:

STRING\$(n," ")

for *n* spaces.

● Tell us who you are. We like to know your Christian name and also it's interesting to know your age and profession. After all, we might reject your program, but if we knew you were a fetlock fettler we'd have been able to send you Obscuresoft's "Fetling fetlocks on the Amstrad CPC464" for review.

Also a telephone number – both home and work – with the correct STD code is really useful, and can save a lot of time.

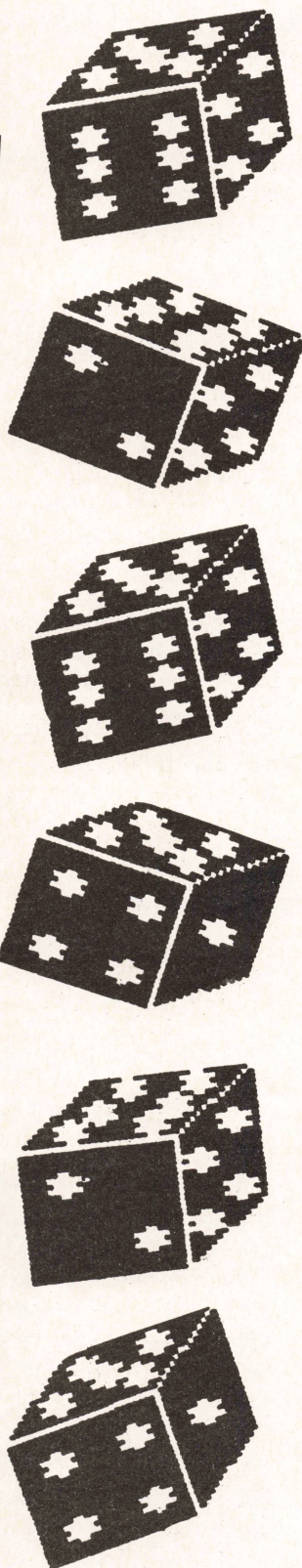
Thus endeth the 18 rules. If you follow these when you submit something to us you'll stand a much better chance of having it published. More importantly, you'll become a far more professional programmer.

And the better you become the more satisfying it is.

RANDOM

By
Aleatoire

REFLECTIONS



THE chance of throwing any number with a die is one in six. What is the chance of throwing a six in three throws? If you think it's 50 per cent then you are a sucker and long may the Ace of Diamonds squirt cider in your ear.

Way back in 16th century Italy gambling with cards, dice and even chess was a major pastime and the more assiduous players knew from experience that you needed four throws to have a roughly even chance of throwing a six.

It was not, however, until about 100 years later, around 1654, that such probabilities were studied and correctly explained mathematically.

Probably the most famous problem was posed by Monsieur de Mere to the French mathematician Blaise Pascal.

M. de Mere knew and accepted that the odds of failing to throw a six are 5/6 and therefore the chance of failing in four throws is:

$$5^4 \cdot 5/6 \cdot 6 \cdot 6 \cdot 6 = 625/1296$$

Consequently the chance of success is 671/1296 as observed in practice, slightly better than evens.

From this de Mere argued that the chance of throwing 12 with two dice (the "Sonnez" from "Sonnez les cloches, le Diable est mort" whenever 12 was thrown in Backgammon) should also be better than evens for 24 throws (since there are 36 possible throws and $36 \cdot 4/6 = 24$).

However de Mere knew from experience that the chance was less than 50 per cent and told Pascal that this revealed a flaw in the mathematical approach. So here is your chance to try the problem and also convince the wife that you didn't just buy the Amstrad to play games.

All you have to do is use the RND function to generate two dice throws and then calculate the number of times the machine manages to throw Sonnez in 24 (or less) throws divided by the total number of trials $\times 100$ per cent.

The following program does this

experiment 100 times in just under 30 seconds and prints the percentage of success:

```
10 RANDOMIZE TIME
20 sonnez=0
30 FOR i=1 TO 100
40 FOR t=1 TO 24
50 d1=INT(RND*6+1)
60 d2=INT(RND*6+1)
70 IF d1+d2=12 THEN sonnez=sonnez+1
80 NEXT t
90 NEXT i
100 PRINT sonnez;"%"
```

If you try it you will find that 100 trials are not particularly conclusive and the mind boggles at how much dice throwing was required to detect the slight difference – certainly many thousands.

Pascal explained that the mathematical proof again depends on calculating the chance of FAILURE to throw the Sonnez in 24 throws, which is $(35/36)$ to the power 24 or:

```
10 c=1
20 FOR i=1 TO 24
30 c=c*35/36
40 NEXT i
50 PRINT c*100;"%"
```

This established the theory of random numbers.

A more modern, though equally famous problem, is that of a submarine captain who is on patrol with only one torpedo left. He expects to separately encounter 10 enemy ships of random size and wants to sink the largest.

This is equivalent to choosing the largest of an unknown set of 10 random numbers when that number is called out and the correct strategy succeeds about once every three trials.

How does the captain choose?
I'll give the answer next month.

ASCII stands for the American Standard Code for Information Exchange. The CPC464 can only deal with numbers. Therefore letters, punctuation marks and symbols have to be stored in memory as numbers. Obviously there has to be a list of which number stands for which symbol and Ascii is the one used in most micros. You can get the chart reproduced at the foot of the page using program I.

Ascii Represents characters in number form.
CHR\$ Gives the character from an Ascii code.
ASC Gives the Ascii code for a character.

```
10 PRINT "Code"; " "; "Character"
20 FOR loop=33 TO 99
30 PRINT " "; loop; "    "CHR$(loop)
40 NEXT loop
50 FOR loop=100 TO 126
60 PRINT loop; "    "CHR$(loop)
70 NEXT loop
```

Program I

```
10 LET variable=68
20 PRINT CHR$(variable)
```

Program II

```
10 LET a=63
20 LET b=6
30 PRINT CHR$(a+b)
```

Program III

Get the facts at your fingertips . . . with the third of our ready reference charts

CHR\$

The keyword CHR\$ allows you to find the character represented by a particular Ascii code. It takes the form:

CHR\$(integer)
PRINT CHR\$(67)

Entering:

gives the letter represented by 67 – which is C. CHR\$ can also take a variable inside the brackets. See Program II.

It can even take an expression, as in Program III.

ASC

The keyword ASC returns the Ascii code of the first character of a string. It takes the form:

ASC(string)

If you want to know the Ascii code for C:

PRINT ASC("C")

will tell you. Note the inverted commas. ASC only returns the code of the first letter of the string.

PRINT ASC("CD")

only displays 67, the code for C. ASC will accept string variables inside the brackets:

10 LET a="CD"
20 PRINT ASC(a)

Notice the inverted commas aren't needed with a string variable.

Code	Character	Code	Character	Code	Character	Code	Character	Code	Character	Code	Character
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	\$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(56	8	72	H	88	X	104	h	120	x
41)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	
45	-	61	=	77	M	93]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o		
48	0	64	@	80	P	96	`	112	p		

Table I: Ascii codes and their associated characters

When looking for a printer...

HAVING shown a passing interest in the Amstrad since it was first announced I was tempted to buy the first issue of your magazine when I saw it in my newsagents.

I also purchased a copy of your competitor, "The official micro magazine", and having made a comparison of the two have decided to take advantage of your subscription offer as your magazine is vastly superior in its features and readability.

I was originally attracted to the Amstrad by its amazing specifications, but have in recent weeks been trying to explore an additional use for the monitor.

Can the Amstrad monitor be connected to a video recorder to improve the quality of its display, particularly with pre-recorded tapes?

I believe most video recorders have a composite video output, although I am not too sure what this means.

If not, would it be possible for me to use the Amstrad with a colour monitor which is suitable for a video recorder, that is one like the Commodore 1701 which has a composite video input.

I notice that the official Amstrad leaflet says that the 6 pin external socket is for RGB and sync, composite video and luminance and sync.

Again I am not too sure what this means. I realise that I will have to use the GT64 to power the computer.

If the above is possible will the monitor have to be returned when going from video recorder to computer, or

will it simply be a case of exchanging plugs?

I hope you can sort this out for me since if I have to purchase a second monitor I will buy the monochrome version of the Amstrad package, but if I can get away with using the same monitor for both applications then obviously the colour system is more attractive. — **J.D. Smith, Fareham, Hants.**

■ The signal for the Amstrad is an RGB colour signal and a B/W composite video.

Most video recorders require a colour composite video — which means the Amstrad isn't suitable for your purpose.

Similarly, a monitor suitable for a video recorder would be equally incompatible. You could manage a B/W signal from the Amstrad on its own, however.

Tips please

I READ your magazine with great interest and I have just got to say how much I enjoyed the Smiley v Grumpies game.

I think that something like a tips page or Top 30 chart should be included. The reviews are good, but I think that ratings, and the price of

the tape, could also be added.

I have just got to recommend The Forest at Worlds Ends by Interceptor Software to your readers.

The graphics are excellent, the vocabulary is large, there are many screens and the plot is good.

I hope to be sending some programs or tips in the future. Keep up the good work. — **Philip Harling (age 12), Southwater, West Sussex.**

Dipswitch solution

I READ Paul Fisher's letter to you with mixed feelings of interest and sympathy. Interest because I wondered if he had anything new to help me with my own problems and sympathy because I knew what he must be going through.

I have a Star Gemini 10X. The following items which I have to contribute I have discovered by experimentation, so they may not be wholly accurate.

Firstly, I will explain a bit about the way in which the CPC464 handles the printer. The computer stores the number of columns on the printer as set by the width command.

It counts the number of characters sent to the printer and then outputs both a line feed and a carriage return when it reaches the preset width.

I imagine that this is necessary for the Amstrad printer, but for more advanced machines which perform their own carriage returns this can be a nuisance as the computer may execute a carriage return in the middle of a line.

Fortunately there is a way to overcome this and the double spacing problem. The printer can be programmed to ignore carriage returns com-

pletely and perform both carriage return and line feed on receipt of a line feed only by setting certain dipswitches. The printer manual will clarify this.

Similarly, it may be programmed to perform both operations on receipt of either character. The default values are the latter.

Therefore when the CPC464 outputs LF and CR the printer double spaces. By setting the dipswitches to the former the printer will only obey the LF, thereby single-spacing.

To get around the width problem you can send a CR every few characters. The printer, of course, will ignore this but the CPC464 will be deceived into believing that the print head is at the first column. — **Malcolm Goris, London.**

Got the jitters

FIRST of all I wish to congratulate you and your colleagues on such a great first edition of your magazine.

I own an Amstrad with the green monitor GT64 and am experiencing a problem with the display. After the monitor has warmed up, the display seems to suffer from a hum-bar which causes the text to oscillate up and down by a few millimetres. It is, needless to say, very annoying.

I have had the original monitor replaced, thinking the fault may be in the PSU, but the fault still persists. Do you think insufficient smoothing may be the problem?

Also, do you have any ideas about screening the Amstrad, as it seems to produce a lot of RFI in the upper VHF region—I measured signals at 180 MHz.

— **Paul O'Neil, Airdrie, Lanarkshire.**

■ It sounds as though it's the

Question of colour

A VERY pleasant surprise to find you on the book stalls, and I look forward with interest to your next issue.

One thing I would like to see covered in a later issue is the matter of which printers, of better quality than the DMP1, can be hooked up to the CPC464.

The query columns of publications similar to yours seem to be so full of little horror stories of cutting wires and of not getting the hash

sign that expert guidance on interfaces, cables etc would certainly be appreciated by readers.

It would also be of great interest to us GT64 monitor owners if some indication can be given by your reviewers regarding what software has a colour selection option or indeed what will produce a satisfactory screen display in monochrome without any modification. — **Alex Rodger, Glasgow.**

micro itself that's at fault – possibly jitter on the synch pulses. Take it back!

As for the RFI – all micros do. Short of encasing it entirely in metal there's not a lot you can do. You'll probably find the colour monitor gives out more than the micro itself!

Disabling reset

At last, a decent, unbiased dedicated computer mag. Well done!

I am the proud owner of an Amstrad colour and, in my capacities as a computer consultant and the organiser of a computer club, I have no hesitation in recommending the Amstrad as one of the best home micros available today.

My only quibbles are the lack of an "OLD" command and the confusion of the ink/pen/mode combinations (considerably less confusing thanks to the Messrs Noels.

Could I ask you to help me out of a small predicament. I am writing demo programs for the Amstrad at work and, although I've solved most of my "interference" problems using the "ON BREAK GOSUB" command, I have forgotten how to disable the keys that reset the computer. Please could you remind me. – Paul R. Seymour, Erith, Kent.

■ Thanks for the praise. The way to prevent reset is POKE &BDEE,&C9

Modem interface

I HAVE just read the first edition of Computing with the Amstrad. Congratulations! It's good to see a magazine published independently of the computer manufacturer.

I would like to see, in your series 'Expansion', an RS232 interface since I wish to hook up a modem to my Amstrad. – Daniel J. Churchward, Crowborough, East Sussex.

■ Mike Cook promises he'll look one up for us.

I ENJOYED the first issue of Computing with the Amstrad very much – a nice balance of items.

I agree with your review of programs from Interceptor Software. Many happy hours have been spent by my family with these products. However, as yet no L9 review. Perhaps later?

I would make the following points based on initial impressions of L9 (Snowball, Lords of Time, etc):

□ Reasonable in concept and no doubt interesting and addictive but why the fancy text which is very hard to read easily? The whole point of a text-based program is to easily read the text and enjoy the actual adventure – not to have to decipher the actual text!

Bigger challenge needed

YOUR new magazine is great, it's more like a book on the Amstrad rather than a magazine.

I have only two quibbles. Firstly could you do some articles on more advanced subjects, as I and my friends who own the Amstrad are taking A level computing and would like some more challenging articles to help our studies?

Secondly, could there be a separate section for useful routines, not mixed in with the letters (as in PCN's microwaves and Random Access pages)?

I have written a small routine which may be of use to Amstrad users. It inverts any of the Ascii character set.

I use it to highlight variables, for example, an "a" which makes the one you want easier to spot.

The routine asks for the Ascii code of the character to be inverted but you could just set the variable in the main

Troubles with too fancy text

□ No graphics (à la Interceptor) to break-up the inevitable boredom of constant text.

□ Seems overpriced at £9.95 compared to other products.

I would be interested in your own views. Also, is it possible to 'fix' the L9 software to default to the standard 'legible' Amstrad text?

If not, I am faced with having L9 products which I am reluctant to use.

I might add that the whole family have problems reading the text (young and old) and none of us need spectacles! – R. Pope, Reading.

■ We totally agree with your comments about the Level 9 text, finding it awful to read. Level 9 feels the script adds to the feel of the game – and

doesn't know of any easy way of restoring it to standard text.

However we can't agree with them being overpriced. On several micros, Level 9 adventures have become the standard against which others are judged and we are sure this will become the case with the Amstrad.

All right, they don't have graphics, but as those of us who prefer radio to television know, the pictures are much better in text adventures.

You'll be glad to know that their latest games, Erik the Viking, Return to Eden and Emerald Isle, do have pictures – and that the latter is three pounds cheaper than the others!

And as you'll see from this issue – Level 9 gets its fair share of reviews.

program. – Anthony Day, Camberley, Surrey.

P.S. I don't know yet how to clear the keyboard buffer, but CALL &BB18 acts like the WHILE-WEND loop shown.

Flushing a buffer

SINCE I bought my CPC464 some two months ago I have been waiting patiently for a dedicated magazine. Now it's here I would like to say it's been well worth the wait.

The content of your first issue was absolutely superb, though I think it was aimed too much at the novice and would like to see a professional column.

In this issue a Mr I. Worth from Glasgow wanted a way to flush the buffer. Well, here are two ways:

□ CALL &BB00. This totally re-sets the key manager, clears the functions key and flushes the buffer. If used in machine code, all registers are corrupted and interrupts are enabled.

□ CALL &BB1B. This doesn't flush the buffer totally. It simply removes the first character in it.

In machine code it can be used to read the keyboard. Register A contains key pressed (if any). The Carry flag will be set if a key is pressed and the registers AF are

```
10 MODE 1
20 SYMBOL AFTER 32
30 INPUT "ASCII CODE OF CHARACTER TO BE INVERSED";CH
40 IF CH<32 OR CH>255 THEN RUN
50 FOR R=0 TO 7
60 BR=PEEK(&A500+(ASC(CHR$(CH))-32)*8+R)
70 C(R)=NOT(BR)-256
80 NEXT R:R=0
90 A1=C(R)
100 A2=C(R+1)
110 A3=C(R+2)
120 A4=C(R+3)
130 A5=C(R+4)
140 A6=C(R+5)
150 A7=C(R+6)
160 A8=C(R+7)
170 SYMBOL CH,A1,A2,A3,A4,A5,A6,A7,A8
180 PRINT STRING$(39,CHR$(C(H)))
```

■ The characters can be set to normal by SYMBOL AFTER 32.

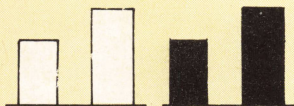
corrupted. — **Mark Woods, Withycombe, Somerset.**

■ Thank you for the routines, unfortunately neither of them is what we wanted — a straightforward buffer flush. If we wanted to, we could flush the buffer by CALL &0 — but, like your first routine, it has too many undesirable side effects!

Filling boxes

I HAVE written a small program to input 12 variables, then print out in form of a bar chart using plot/draw commands and as part of a larger program.

This gives rectangular transparent boxes but in order to see the display more clearly I would prefer to have the display filled with pen ink, such as:



but don't know how to do this on the CPC464.

I don't really want to use a fill routine since this would take too long. Can you therefore advise if it is possible to use say CHR\$(143) statement and how? — **S.G. Squires.**

PS. Thanks for the excellent magazine.

- This month's graphics article by Michael Noels gives a straightforward way to do it:
- Define a window to the appropriate size.
- Set paper to the colour you wish to fill it with.
- Clear that window.

You'll then have a rectangle of the correct size filled with the colour you want. This, of course, makes the use of CHR\$(143) redundant.

Decimal to binary

I WAS delighted to find a program to convert decimal to binary in your January issue as

I don't know how to do this.

Using my Amstrad, I found that decimal 3 is 00000011 and 0 is 00000001. I know how to convert binary to Hex and found that I had 1 in hex. I then used my BBC Micro to convert &1 to 1 and therefore proved that 0=1!

I asked a friend, and he said just put " " at the end of line 120 and 0=0. — **Frank Marsden, Leicester.**

■ Yes, there was a little hiccup wasn't there? Of course 0 should be 00000000 in binary, and if you look at the number the program gave you for 0 it was a nine bit byte!

The problem was caused by the value for 0 (00000000) overprinting the value for 3 (00000011) — but displaced one character to the left. This leaves the last 1 of the 3 showing after the eight bits of the 0.

We just didn't expect anyone to want to find the binary value of 0! Still, it was a bit careless. As your friend said, adding a space to the end of line 120 will cure things.

Problem solved

I AM writing to answer some of the questions that have been asked by your readers.

David Ingles asked about a modem for the Amstrad. I can safely say that Protek are to release one some time this year.

The second answer is for Julie Bayliss. I am sure that

she will be pleased to know that Skywave Software are releasing Fig-Forth for the Amstrad. — **Martin Murray, Ellesmere Port, South Wirral.**

■ One or two companies are producing RS232 interfaces for the Amstrad and, if you read page 00, you'll find something that should interest you.

Skywave has released a Fig-Forth on cassette. Even better news is that they're working on a ROM based implementation that should be multi-tasking.

I like it . . .

I CAN hardly believe my double good fortune!

First of all, as a complete novice in the computing field, I had the luck to decide on the superlative CPC464 and now YOU follow up with what must surely be the best and most comprehensive computer magazine on the market.

It's so packed with useful ideas and information that I'm like a glutton, avidly trying to hurry and absorb the multi-course, gourmand meal you've provided me with.

Your editorial and production staff, as well as all the contributors deserve the highest praise and it may be invidious to select the Michael Noels' Graphics article for special praise. This is only because now, for the first time, I at last understand how the

Amstrad colour system works!

Nice going fellows. Keep it up! — **Peter Smith, Bournemouth.**

Scrolling along

I'M over the moon with your magazine. When my wife brought it in, it was just like getting a Christmas Box early. You will save me money now, for all I need is one magazine — YOURS. It is just what I have been waiting for.

Regarding your program Scroller, I tried it and found it to be a bit long for what it does. So I put on my thinking cap and came up with using MID\$ to do the same job, which resulted in the short program below. I hope you don't mind, but I have called it Another Scroller.

```
5 REM ANOTHER SCROLLER
10 REM BRIAN GADSDEN
15 MODE 1
20 INPUT "WHAT MESSAGE";A$
25 INPUT "WHICH LINE";A
30 IF A<1 OR A>25 GOTO 25
35 CLS
40 A$="          "+A$+" "
45 i=1
50 LOCATE 15,A:PRINT MID$(A$,i,10)
55 FOR t=1 TO 500:NEXT
60 i=i+1
65 d$=INKEY$:IF d$="" THEN
70 ELSE 15
70 IF i=LEN(A$) THEN 45 ELS
E 50
```

In the MID\$ function, I used the second integer expression to vary the length of the sub-string to be shown across the screen. — **Brian Gadsden, Sunderland.**

■ Many thanks for the program, which is nice but doesn't do the same job as Scroller.

Scroller not only displays all of the message at the same time it also has a "wrap-around" effect which ensures that as a letter disappears at one end it appears at the other. Your program does neither, which is why it's so much shorter. It's good, though!

Computing with the AMSTRAD Postbag

WE welcome letters from readers — about your experiences using the CPC464, about tips you would like to pass on to other users . . . and about what you would like to see in future issues.

The address to write to is:

**Postbag Editor
Computing with the Amstrad
Europa House
68 Chester Road
Hazel Grove
Stockport SK7 5NY**

Industry standard

AS a relative newcomer to computing – and the CPC464 – I would appreciate some guidance on compatible printers.

The Amstrad printer has had some poor reviews and various comments have been raised over graphic/screen dumping on other printers.

Are there any printers you would recommend for the Amstrad, or is it better to wait a while and see what develops? What about interfaces or leads – who supplies these and at what cost?

Any advice would be greatly appreciated. Congratulations on your new mag – it's excellent. – K. Jones, Ulverston, Cumbria.

■ We must agree with those who weren't too impressed with the Amstrad printer – certainly none of us use it.

You ought to look for a printer that is Epson compatible. Epson really are the Rolls Royce of dot matrix printers, and their system of control codes has become the de facto industry standard.

So an Epson compatible printer should have the necessary Centronics-type

interface and a host of facilities such as underlining, paging, condensed characters and so on.

If you want to do satisfactory screen dumps though, you'll need a printer that also supports bit or graphic image mode. Check for this – describing a printer as "Epson compatible" does not necessarily mean it has this.

The 7 bit hurdle isn't all that much of a problem as far as screen dumps are concerned.

Quite a few printers fall into this category – the Epson FX80, and the considerably cheaper Taxan Kaga are but two.

As for the lead – the person who sells you the printer will sell you a suitable cable.

Spectrum switch

AT last a magazine for the CPC464, and a very good one if I may say so is your first issue. For the first time in my life – I am 61 by the way – it has caused me to write a letter of appreciation to a magazine. I found it very informative and adult.

You are on the right track, and no mistake.

You ask for suggestions for

future articles. May I suggest that as the majority of games and written programs are for the Spectrum and I should guess that most of your readers will be ex-Spectrum types, with books etc for this computer, could you interpret the Sinclair Basic into Amstrad Basic in an article that simplifies it for people like me.

Wishing you all every success. – D.R. Cooke, Hove, Sussex.

■ A good point, but unfortunately (or fortunately, depending on how you look on it) no one here has really used Spectrum Basic.

Is there any reader out there who thinks he could write a lucid article on translating from Spectrum to Amstrad Basic?

Up tempo 464?

WHAT a fantastic magazine you have. Please keep up the good work.

I am writing to ask if there are plans for the Amstrad CPC464 to be used for computer music, with an add-on-music keyboard etc.

Examples I was thinking of were the CX5M Yamaha music computer and the Music

500 synthesiser for the BBC B. – Jeff Cuckson, Newtownabbey, N. Ireland.

■ We haven't heard of anything musical yet – but would bet quite a bit that there'll be plenty soon.

Is there anyone out there producing anything?

Seeing the light

FOR many months I have been purchasing micro magazines in an effort to get a clear picture of many needs and to find a regular magazine that was actually compatible to the micro they purport to support.

I have not been successful until now and at last I think you have the magazine I've been after.

I do not yet have a micro but I am now at ease with the intricate nature of at least one after reading the first issue of Computing with the Amstrad, so much so that my purchase will be in this direction.

I have also decided to take the magazine subscription for the next 12 months and no doubt I will have the dust cover before the micro arrives.

– D.M.G. Lewis, Newport, Gwent.

When typing in pays off

AT last, a magazine for the Amstrad, sorry, the CPC464. No longer will I have to envy the Commodore and Sinclair users their own magazines.

Looking through the first issue, I can only offer congratulations. The CPC464 is the first computer I've owned, in fact the first computer I've laid hands on.

That is why I think you are to be congratulated, for the articles start from scratch. My only complaint is that it arrived six months late.

Nice as it is, I don't wish to know what some genius can do with graphics and sound, not yet anyway.

The fact that the sound capabilities are manna from

heaven for some, doesn't excite us first timers, unless someone has the will to literally start as you have, from the beginning.

Also for the last six months I've owned but have not been able to use the machine for what I had in mind. The 30 games on sale at its release did little to help someone who had bought the B/W monitor version.

But now Amsword has arrived every spare moment over the Christmas the Quen-Data daisywheel printer I've bought has been chattering away on anything the family needs in the way of stationery.

By now you've probably gathered that I'm not only a

first timer but also an old timer, at least I've been retired for a few years.

But that doesn't mean that I'm rusty, and I am thirsting for some original programs that can justify this expensive "ornament". Don't get me wrong, with this set up you can keep your QLS but dear editor, please spare us too many of those games.

It may seem odd to congratulate you on the really "Basic" articles that have appeared so far and to be negative about the standard of programs that have appeared elsewhere for the CPC464, at least they taught me to type a program.

I look to you to provide me

with much more, and so do many other readers I'm sure.

Finally, is there a CPC464 user club in the Bristol area? If there is I would like to hear from them. Again congratulations, I'm reading your "first" articles repeatedly. – D.S. Thomas, Bristol.

■ Thanks for the praise – and don't worry we're not going to be just a games magazine.

However many people do enjoy typing in games – and not just for playing. As you point out, you can learn a lot from them.

Also the prospect of tailoring a particular game to your own requirements is far less daunting than starting one from scratch.

Amstrad CPC464

Speech Synthesizer

The dk'tronics Amstrad speech synthesizer and powerful stereo amplifier uses the popular SLO/256 speech chip and has an almost infinite vocabulary. It is supplied with a text to speech converter for ease of speech output creation. Everything you wish to be spoken is entered in normal English, without special control codes or characters, it is therefore extremely easy to use. The voicing of the words is completely user transparent and the computer can carry on its normal running of a program while the speech chip is talking. The speech output from SLO/256 is mono and directed to both speakers.

Stereo Output

To utilise the Amstrad stereo output on the back of the computer, the interface has a built in stereo amplifier, this gives all sound output a totally new dimension and greatly improves the sound quality and volume over the computer's internal speaker. Any sound that previously came out of the mono speaker will now be sent out via the interface in stereo. All programs that use the sound in anyway (i.e. commercial software) will now output through the interface, which is fitted with volume and balance controls.

Speech Synthesis

The Amstrad speech synthesis utilises parts of the spoken word known as allophones. These are actual sounds that go to make up speech. The SP0256 allophone speech synthesis technique provides the ability to synthesize an almost unlimited vocabulary. Fifty-nine discrete speech sounds (allophones) and five pauses are stored in the speech chip's internal rom.

Text to Speech

Although there are only 26 letters in the alphabet, letters have a totally different sound when used in different words. For example, The 'a' in 'Hay' is much longer and softer than in 'Hat'. When you speak you automatically make adjustments because you know just how a word should sound. Not quite so easy with a computer.

The machine code software is mainly developed to this mode of operation. 3.5K is used for tables which contain the rules & exceptions to the rules of the English Language.

e.g. I before E except after C) This therefore allows the user to enter words to be spoken in normal English.

Speakers

Supplied with the Speech Synthesizer are two high quality 4" speakers these have been designed to compliment the Amstrad Computer. They are fitted with 1 metre of cable and can be positioned for the best stereo effect. The synthesizer interface fits neatly on to the rear of the computer. It has a through connector to enable other interfaces (e.g. Disc Drive) to connect to the rear of the synthesizer for ease of expansion. Please send S.A.E. for a copy of the instruction manual which will give full and comprehensive details.



New Basic Commands

There are 8 new Basic Commands which control all the functions of the interface. Making the Synthesizer very easy to use. You can even control the speed at which it will talk to you. Or use the synthesizer to create sound effects like a fourth sound channel.

10 PRINT " 'AMSTRAD' "

The above is an example of the Syntax for entering speech into the computer and shows how simple it is to use.

The instruction book gives comprehensive details and examples of how to use the interface both from machine code and basic.

How to Order

The Amstrad Speech Synthesizer costs only £39.95. You can obtain your synthesizer through any good computer store or by completing the order form and returning it to: dk'tronics Limited, Shire Hill, Saffron Walden, Essex. OR by telephone quoting your Barclaycard or access number. Orders normally despatched within 24 hours.

Please rush me

.....[QTY] Amstrad Speech Synthesizer at £39.95 + £1.25 p&p
I enclose cheque/PO/Cash for Total £.....
or debit my Access/Barclaycard No.

Signature.....

Name.....

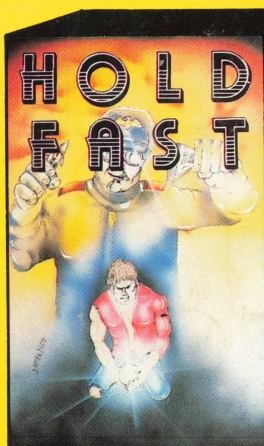
Address.....

dk'tronics

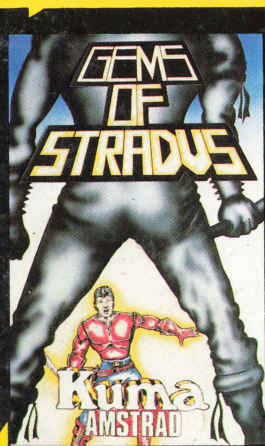
Saffron Walden, Essex CB11 3AQ
Tel: (0799) 26350 10 lines

the only choice

Kuma AMSTRAD CPC464 software



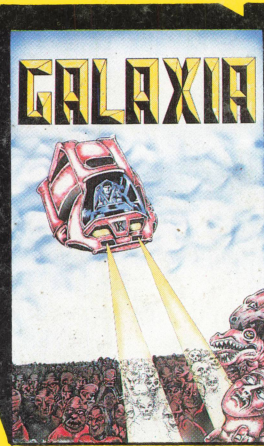
Holdfast



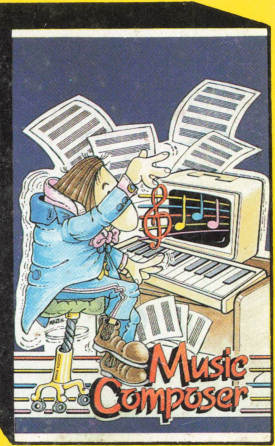
Gems of Stradus



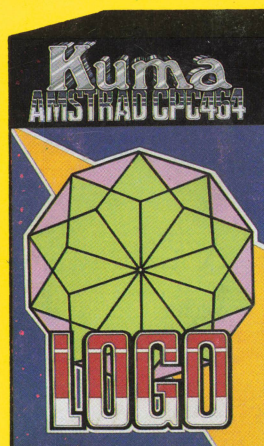
Star Avengers



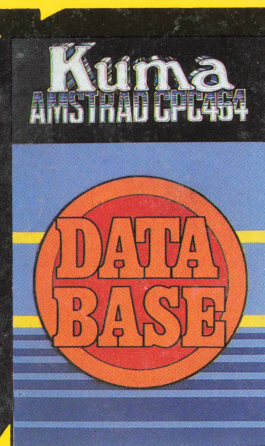
Galaxia



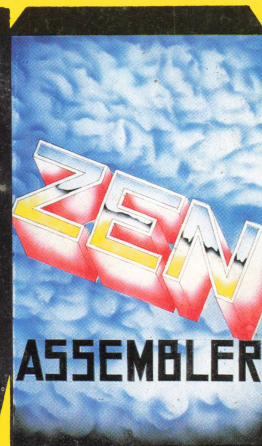
Music Composer



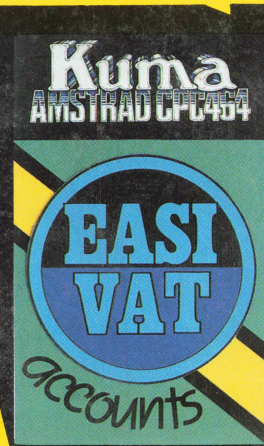
Logo



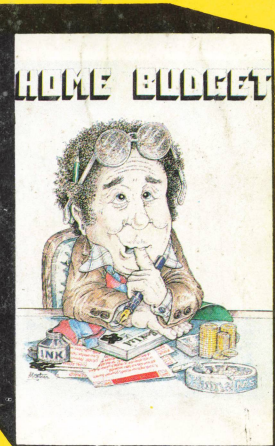
Database



ZEN Assembler

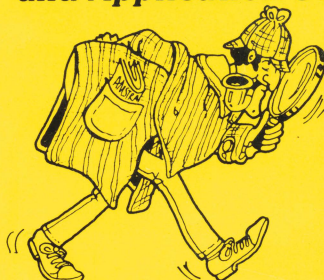


EASIVAT



Home Budget

An outstanding selection from Kuma's rapidly expanding range of Entertainment and Application Software for the Amstrad CPC 464 Micro-computer.



BOOK ● The Amstrad CPC 464 Explored.

This superb book is designed to let every CPC 464 user, at whatever level, get the most from his computer. After an introductory section on the special Basic features, the book looks in depth at the excellent sound and graphic facilities.

Now available from selected branches of Co-op, Granada, LASKYS and John Menzies

Kuma Computers Ltd., Unit 12, Horseshoe Park, Horseshoe Road, Pangbourne, Berks RG8 7JW.

Please send full catalogue on Amstrad CPC464 products.

Name

Address

..... Phone.....

I own an Amstrad CPC 464 computer ☐

Trade Enquiries Phone 07357-4335

Visitors wishing to call at our Pangbourne Manufacturing and Distribution Centre are advised to phone 07357-4335 first for an early appointment.